Received April 2, 2022, accepted May 5, 2022, date of publication May 13, 2022, date of current version May 19, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3174859

A Detection Model of the Complex Dynamic Traffic Environment for Unmanned Vehicles

SHIJUAN YANG, LI GAO, AND YANAN ZHAO
School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China

Corresponding author: Yanan Zhao (zyn@bit.edu.cn)

This work was supported by the National Key Research and Development Program of China under Grant 2018YFB0105205-02, Grant 2017YFC0804808, and Grant 2017YFC0804803.

ABSTRACT It has always been an important and arduous task to detect the complex dynamic traffic environment, especially for unmanned driving. Although the existing advanced detection models have reached the speed requirements for detection, the detection accuracy needs to be further elevated to improve the unmanned driving's safety. How to balance the accuracy and speed of detecting the complex dynamic traffic environment is still the primary problem to be solved for unmanned vehicles. Therefore, this article proposes a detection model of the complex dynamic traffic environment for unmanned vehicles by following the framework idea of YOLOv3. Firstly, we regard MobileNetv3 as the backbone and replace the traditional convolution with the depthwise separable convolution in the whole model to reduce the number of parameters and calculations. Secondly, in enhanced feature fusion layers, we perform the multi-scale fusion of four feature maps by the compress-and-expand module, the SPP module, and the cross-layer bidirectional module of feature fusion to improve the locating accuracy and reduce false detections. Thirdly, we add an IoU loss to improve the accuracy of model regression. Then, we employ the improved clustering algorithm to re-cluster anchor boxes, reducing the time overhead while improving the clustering accuracy. Finally, we compare the proposed model with other advanced detection models in the processed BDD dataset and the KITTI dataset. We verify that the mAP of the proposed model improves notably without loss of detection speed, the number of parameters and calculations decreases dramatically, and the proposed model exhibits a more superior performance.

INDEX TERMS Complex dynamic traffic environment, detection model, unmanned vehicles, YOLOv3.

I. INTRODUCTION

For unmanned vehicles, the complex dynamic traffic environment is composed of all the moving elements that may affect the driving of the unmanned vehicles themselves, just like various types of vehicles in the lane, pedestrians, and riders [1]–[3]. It has always been an important and arduous task to detect the complex dynamic traffic environment. It affects the planning decision, control execution of unmanned driving, and ultimately the safety of unmanned driving.

Contrasted with other sensors, machine vision has an excellent performance in classifying different vehicles, pedestrians, and riders [4]. Therefore, target detection technology based on machine vision is widely used to classify and locate moving elements in the complex dynamic traffic environment [5]–[7]. However, the target detection based on machine vision has many problems in practical situations.

For the classification and locating of moving elements (targets) in the complex dynamic traffic environment, the difficulties of target detection technology based on machine vision mainly focus on the impact of the characteristics' diversity of pedestrians and riders, scales' diversity of vehicles, pedestrians, and riders, environmental factors, and the mutual occlusion between targets. In order to solve the above-mentioned difficulties, many scientific research institutions and scholars have conducted extensive and in-depth research in recent years.

With the development of machine vision technology, the target detection technology has successively undergone the frame difference method, the optical flow method, the background difference method [8]–[10], the template matching method [11], and the statistical learning method [12], [13]. Whereas the seemingly well-developed detection model based on statistical learning faces the inability to balance the relationship between high-quality detection and a large volume of time-consuming calculations, it cannot meet the

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

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

detection needs of unmanned driving for accuracy and speed. Development and application of deep learning break through the bottleneck of the target detection based on statistical learning [14]–[16]. At present, the mainstream general target detection technologies based on deep learning mainly consist of the method based on region proposal generation such as R-CNN series and the methods based on regression such as YOLO series and SSD series.

The detection speed of the detection model based on the regression method is generally faster than the detection model based on region proposal generation. However, the detection accuracy is worse than that of the detection model based on region proposal generation [17]. In order to improve accuracy, neural network models gradually deepen and widen, such as YOLOv3 [18] and YOLOv4 [19]. Although the detection accuracy of YOLOv3 and YOLOv4 is comparable to or even better than the detection accuracy of models based on region proposal generation, they still need to be further improved. As the network models deepen and widen, the number of parameters and calculations is considerable, which is extremely unfavorable for the detection models. How to balance the detection accuracy and speed of the complex dynamic traffic environment is still the primary problem to be solved for the realization of unmanned vehicles [20]. Moreover, the YOLOv3 model also has some problems that include poor network clustering, inaccurate target locating, and false detection, which seriously threaten the safe driving of unmanned vehicles [21], [22].

In response to the above-mentioned problems, we choose the framework of YOLOv3 that is a typical representative of the current advanced target detection models to build a detection model of the complex dynamic traffic environment for unmanned vehicles. Firstly, MobileNetv3 that introduces the attention mechanism to the inverted residual block is in place of the backbone network of YOLOv3. Furthermore, we employ the depthwise separable convolution instead of the traditional convolution to reduce the number of parameters and calculations. Secondly, in enhanced feature fusion layers, the top feature map extracted by the backbone passes through the designed compress-and-expand module and the SPP module to enhance the fusion of features and reduce the number of parameters and calculations continuously. In order to improve the accuracy of the target locating and avoid false detection in a complex dynamic traffic environment, we design the cross-layer bidirectional module of feature fusion to realize the multi-scale fusion of four effective feature maps and make full use of shallow-layer and deep-layer information. Thirdly, we add an IoU loss in the loss function and take advantage of the form of binary cross-entropy to predict the center offset loss of the bounding box to improve the regression accuracy of the detection model. Moreover, we improve the k-means clustering algorithm in YOLOv3 and adopt the improved clustering algorithm to re-cluster the anchor boxes. The improved clustering algorithm enables to avoid the selection randomness of the initial clustering center, the influence of noise and interference from external factors, and a lot of run-time overhead. Finally, we conduct the experiments in the processed BDD dataset and the KITTI dataset and compare the proposed detection model with other models to verify the superior performance of the proposed detection model.

II. RELATED WORKS

A. TRAFFIC TARGET DETECTION BASED ON DEEP LEARNING

After 2010, the field of target detection has entered a freezing period, and there has hardly been any innovative development. The success of the AlexNet [23] allows many researchers to see new opportunities and marks the advent of the era of deep learning [24], [25].

As a two-stage model, the detection model based on region proposal generation first selects a proposal box for the input image and then classifies and locates the proposal box to get the final detection results [26]. R-CNN [27] regarding AlexNet as the backbone network pioneers the application of CNN in target detection. Although the detection accuracy of the R-CNN model is better than that of the traditional target detection method, the recurring proposal boxes increase a lot of calculations and cause the detection speed to slow down. To overcome this problem, He *et al.* propose the SPP-Net (spatial pyramid pooling network) [28]. The SPP-Net only performs feature mapping once for the entire detection target, thereby reducing the detection time. Fast R-CNN [29] combines the idea of the SPP-Net and improves the detection speed and accuracy once again. The Faster R-CNN [30] generates candidate regions by using the Region Proposal Network (RPN), which truly realizes the end-to-end training of the target detection. R-FCN [31] takes ResNet [32] as a feature extraction network to improve the effect of feature extraction and classification. Subsequently, the Mask R-CNN [33] and the Cascade R-CNN [34] also continue to solve the shortcomings of the previous models, but they also bring new problems. The problems such as large model scale and slow detection speed have still existed.

As a single-stage model, the detection model based on regression omits the generation stage of the candidate region and can directly obtain the target's classification and position coordinates. In response to the widespread problem of poor real-time performance in two-stage models, Redmon *et al.* propose the YOLOv1 model that is the first single-stage network [35]. The YOLOv1 model treats the target detection task as a regression problem. As long as it processes the input image once, it can get the position and the class of targets simultaneously. Since YOLOv1 does not generate candidate regions, it has a fast detection speed where YOLOv1 greatly exceeds the two-stage models. Nevertheless, YOLOv1 produces more locating errors that result in low overall detection accuracy. Basing on YOLOv1, W. Liu *et al.* propose the SSD (Single Shot MultiBox Detector) [36]. SSD applies an RPN-based mechanism and end-to-end regression to improve the detection accuracy, but the detection speed is slightly slower than that of the YOLOv1 model. YOLOv2 [37] predicts the bounding box by anchor boxes and takes the more efficient Darknet-19 as the backbone network.

YOLOv3 regards the better Darknet-53 as the backbone network to realize a faster detection speed. By means of FPN [38], it performs the detection task on the feature maps with three different scales at three different positions to improve the detection effect of the network effectively. YOLOv3 is a typical representative in the YOLO series as well as the most widely used anchor-based one-stage detection model. Later YOLOv4 continuously improves on the framework of YOLOv3 by adopting CSPDarknet53 and PAN [39]. Consequently, we choose the framework of YOLOv3 as the basis to build a detection model of the complex dynamic traffic environment for unmanned vehicles.

For target detection, a dataset with strong applicability is also an indispensable requirement as support in addition to a powerful network framework. Aiming at the traffic environment, some core autonomous driving companies and major institutions, such as Alphabet-Waymo, Uber, Tencent Baidu, *etc.*, provide a large number of training and testing datasets required for detection. Meanwhile, some non-profit organizations and colleges also offer some marked training datasets freely. Prevalent datasets about traffic targets include KITTI [40], Cityscapes [41], ApolloScape [42], Mapillary [43], and BDD100K [44]. The Berkeley Diverse Drive (BDD100K) dataset is more diverse than several other public datasets in terms of the number of images, city samples, backgrounds, weather conditions, and lighting conditions [45]. Its samples are more consistent with complex dynamic traffic scenes. In the article, we choose the BDD100K dataset and the KITTI dataset to train and evaluate the detection model of the complex dynamic traffic environment for unmanned vehicles.

B. FRAMEWORK OF THE YOLOV3 MODEL

The framework of the YOLOv3 model mainly includes the backbone, enhanced feature fusion layers, and YOLO heads. Fig. 1 displays the main framework of the YOLOv3 model.

The backbone is responsible for extracting targets' features. The YOLOv3 model extracts targets' features in the image by the structure of Darknet-53. The images in the dataset are normalized to a size of 416×416 and sent to the detection model. Darknet-53 contains a large volume of residual blocks composed of 1 × 1 and 3 × 3 convolution kernels in the framework. The 3 × 3 convolutional layer is in charge of increasing the number of channels to extract features, and the 1 × 1 convolutional layer is in charge of adjusting the number of the channels. The backbone of the YOLOv3 has 52 convolutional layers. It enables to extract three effective feature maps of 13 × 13, 26 × 26, and 52 × 52 [68].

The YOLOv3 model regards FPN as the enhanced feature fusion layers. FPN enlarges the small feature map to the same size as the feature map of the previous layers by upsampling. As the number and scale of the final feature maps change, the size of the anchor boxes also needs to be adjusted accordingly. The YOLOv3 model clusters the anchor boxes in the training set by using the k-means clustering algorithm and selects representative anchor boxes based on the clustering results. It requires nine sizes of anchor boxes in total.

The YOLOv3 model includes three YOLO heads. YOLO heads correspond to three effective feature maps of 13 × 13, 26 × 26, and 52 × 52 to realize multi-scale target detection.

III. DETECTION MODEL OF THE COMPLEX DYNAMIC TRAFFIC ENVIRONMENT FOR UNMANNED VEHICLES

Learning from the design ideas of the YOLOv3 framework, we propose a detection model of the complex dynamic traffic environment for unmanned vehicles. The proposed detection model contains feature extraction layers (backbone), enhanced feature fusion layers, and YOLO heads. The structure of the proposed detection model is visible in Fig. 2.

A. BACKBONE

An excellent backbone directly refers to the accuracy of network recognition subsequently. At present, neural networks have be more and more complex and deeper and deeper. Despite the accuracy of the network model has improved, the number of parameters and calculations becomes more and more. The considerable number of parameters and calculations is unfriendly to detecting the complex dynamic traffic environment for unmanned vehicles. In [46]–[48], the MobileNetv3 is the lightweight network and makes the network convolution process more efficient as the backbone. It is able to reduce the number of model parameters and calculations on the premise of a small decrease in accuracy. Therefore, we extract features of dynamic environmental targets by the MobileNetv3 [49] announced by Google in 2019. Table 1 shows the overall structure of the Mobilenetv3. The MobileNetv3 continues to use the depthwise separable convolution, the inverted residual block, the linear bottleneck structure, *etc.*, while adopting the SE module and h-swish function. Compared with previous MobileNetv1 [50] and MobileNetv2 [51], the performance and speed of MobileNetv3 improve to an extent. In the proposed detection model, we take the rest of MobileNetv3 as the backbone network after removing the pooling layer and the convolutional layers. Since the shallow-layer feature maps have rich location information and the deep-layer feature maps have rich semantic information, the backbone extracts four feature maps to improve the detection accuracy.

1) DEPTHWISE SEPARABLE CONVOLUTION

The prominent advantage of the MobileNet series is the use of the depthwise separable convolution to reduce the number of parameters and calculations [53]. The depthwise separable convolution is able to increase the detection rate without significant changes in the detection accuracy. It defines two independent layers in Fig. 3, the lightweight depthwise convolution for spatial filtering and the pointwise convolution for feature generation. The characteristic of depthwise convolution is that the number of channels of the convolution

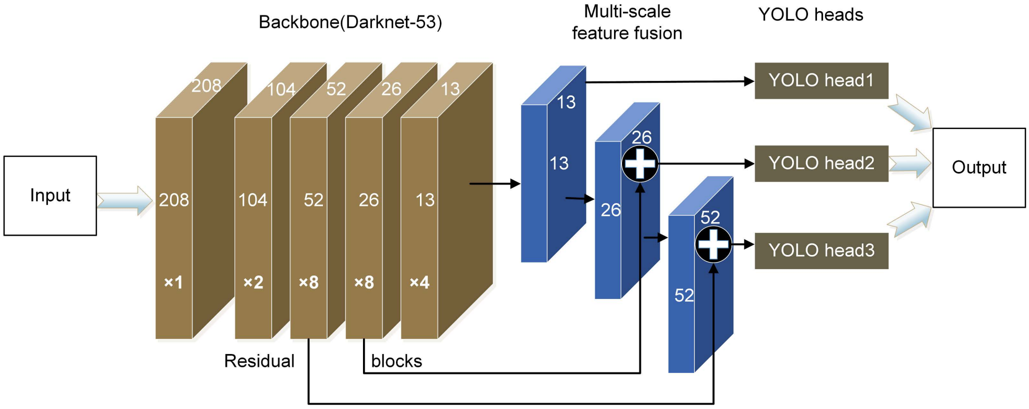


FIGURE 1. Framework of theYOLOv3 model.

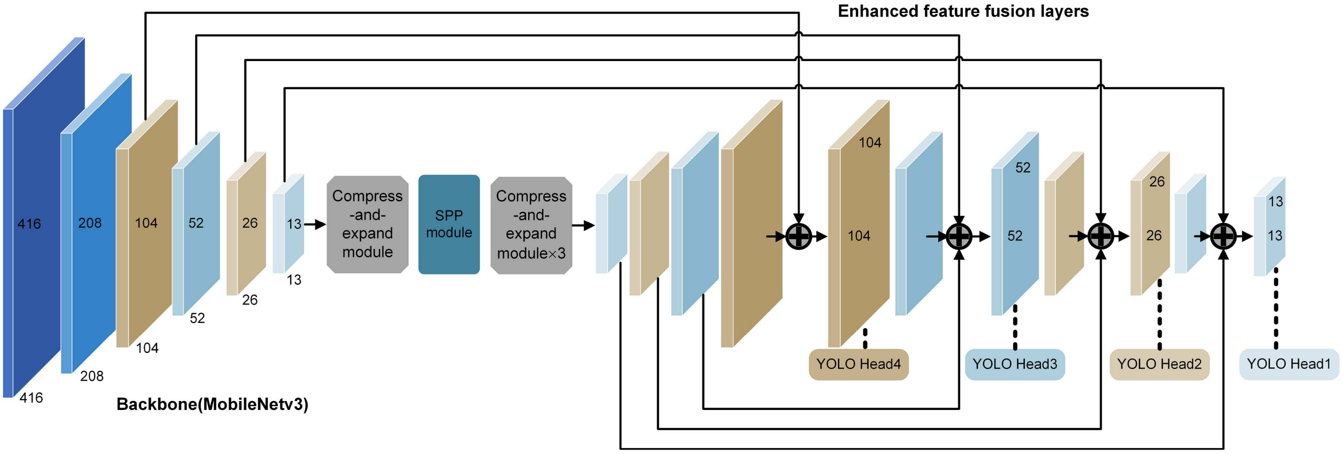


FIGURE 2. Structure of the proposed detection model.

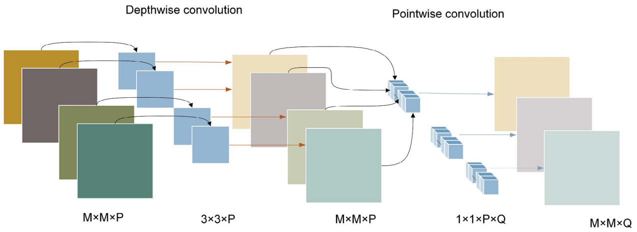


FIGURE 3. Depthwise separable convolution.

kernel is 1. The pointwise convolution is essentially a 1 × 1 convolution kernel, and the number of channels is equal to that of the output feature map. As we know, a large number of parameters come from the 3 × 3 convolution kernel in YOLOv3. Taking a 3 × 3 convolution kernel as an example, we compare the depthwise separable convolution with the traditional convolution to illustrate that the number of parameters and calculations of the depthwise separable convolution significantly decreases in Table 2. The feature map of the input is M × M × P, the stride is 1, the convolution kernel is 3 × 3 × Q, and the feature map of the output is M × M × Q.

Basing on the advantages of the depthwise separable convolution, we replace the traditional convolution with the depth separable convolution in the proposed detection model of the complex dynamic traffic environment for unmanned vehicles.

2) INVERTED RESIDUAL BLOCK WITH SE STRUCTURE

The main structural block of Mobilenetv3 is the inverted residual block with the SE (Squeeze-and-Excite) structure. For the usual residual block, it compresses the number of channels of the feature map by a 1 × 1 convolution kernel firstly. Then, it passes through a 3 × 3 depthwise convolution layer. Finally, the usual residual block expands the number of channels by a 1 × 1 pointwise convolution layer. In short, the usual residual block expands to the previous number of channels after compressing channels of the feature map. The depthwise convolution layer extracts a few features in the usual residual block because the number of input channels restricts the network to extract features. In order to increase

TABLE 1. **Overall structure of the MobileNetv3. S is the stride. SE means whether there is a Squeeze-and-Excite in that block. NL means whether the block use HS (h-swish) or RE (ReLU).**

Input	Operator	S	Exp size	Output	SE	NL
416×416×3	Conv2d	2	-	16	-	HS
208×208×16	Bneck,3×3	1	16	16	-	RE
208×208×16	Bneck,3×3	2	64	24	-	RE
104×104×24	Bneck,3×3	1	72	24	-	RE
104×104×24	Bneck,5×5	2	72	40	✓	RE
52×52×40	Bneck,5×5	1	120	40	✓	RE
52×52×40	Bneck,3×3	1	120	40	✓	RE
52×52×40	Bneck,3×3	2	240	80	-	HS
26×26×80	Bneck,3×3	1	200	80	-	HS
26×26×80	Bneck,3×3	1	184	80	-	HS
26×26×80	Bneck,3×3	1	184	80	-	HS
26×26×80	Bneck,3×3	1	480	112	✓	HS
26×26×112	Bneck,3×3	1	672	112	✓	HS
26×26×112	Bneck,5×5	2	672	160	✓	HS
13×13×160	Bneck,5×5	1	960	160	✓	HS
13×13×160	Bneck,5×5	1	960	160	✓	HS
13×13×960	Conv2d	1	-	960	-	HS
13×13×960	Pool, 7×7	1	-	-	-	-
1×1×960	Conv2d 1×1	1	-	1280	-	HS
1×1×1280	Conv2d 1×1	1	-	k	-	-

TABLE 2. **Number of parameters and calculations about the depthwise separable convolution and the traditional convolution.**

Convolution	Number of parameters	Number of calculations
Traditional convolution	3×3×P×Q	M×M×3×3×P×Q
Depthwise separable convolution	3×3×P+P×Q	M×M×(3×3×P+P×Q)

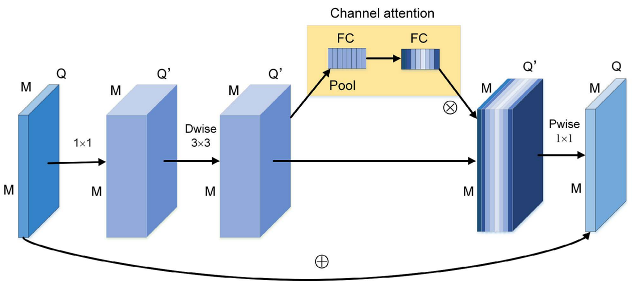


FIGURE 4. **Inverted residual block with the SE structure.**

the number of channels and obtain more features, the inverted residual block first expands channels by a 1 × 1 convolution kernel. Moreover, the inverted residual block introduces the SE structure in Fig. 4. The SE structure mainly learns the correlation between channels to filter out the attention to the channels. It processes the feature map to obtain a one-dimensional vector with the same number of channels. The one-dimensional vector is the evaluation score of each channel. The SE structure applies the scores to the corresponding channels afterward to stimulate the useful channels and suppress the useless channels [54]. Despite the SE structure increases the number of calculations slightly, the effect of the inverted residual block with the SE structure is better than that of the usual residual block.

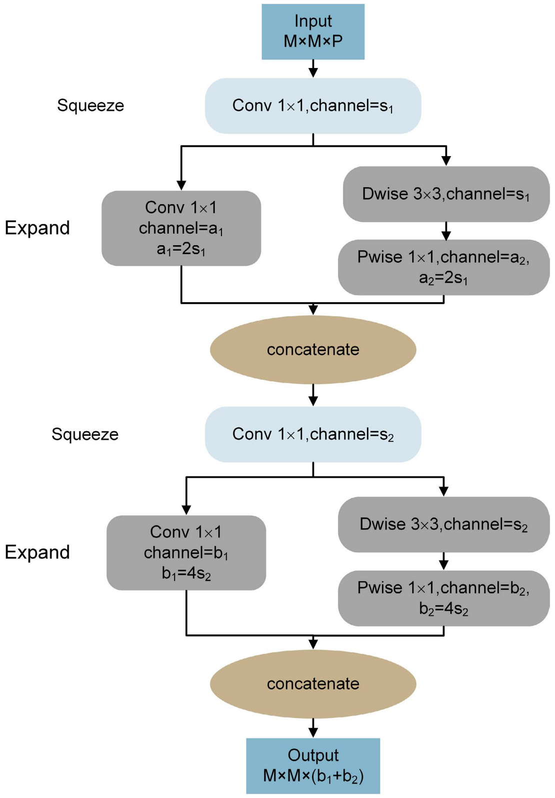


FIGURE 5. **Structure of the compress-and-expand module.**

B. ENHANCED FEATURE FUSION LAYERS

In YOLOv3, the structure of FPN can fuse the feature map with strong low-resolution semantic information and the feature map with rich high-resolution spatial information from top to bottom. In the proposed detection model, enhanced feature fusion layers continuously strengthen the top features extracted by the backbone through the compress-and-expand module and the SPP module. After that, the obtained feature map performs multi-scale fusion with the other three feature maps from top to bottom and from bottom to top in the cross-layer bidirectional module of feature fusion. The process is able to solve the problems of insufficient use of shallow-layer information and loss of deep-layer information, thereby improving the accuracy of target position and reducing false detections in the complex dynamic traffic environment.

1) COMPRESS-AND-EXPAND MODULE

In Fig. 5, the compress-and-expand module compresses the channels of the input M × M × P through a convolution kernel with the size of 1 × 1 × s_1 and then expands the channels of the feature map by the 1 × 1 convolution and the 3 × 3 depthwise separable convolution simultaneously. The number of channels of the convolution kernels is a_1 and a_2 respectively. We obtain a feature map with a size of M × M × $(a_1 + a_2)$ after concatenation. At last, the compress-and-expand module compresses, expands, and concatenates feature maps again. As shown in Fig. 5, $a_1 = a_2 = 2s_1 = 1/4P$,

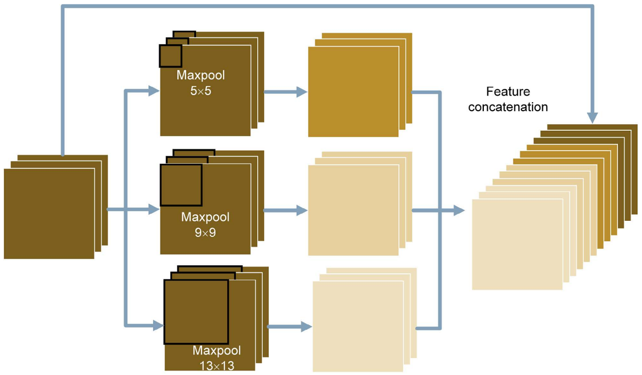


FIGURE 6. **Structure of the SPP module.**

and $b_1 = b_2 = 4s_2 = 1/2P$. The convolution kernels with different sizes mean the receptive fields with different sizes, and the final concatenation means the fusion of features with different scales. Moreover, the number of channels has shrunk exponentially before expanded, thus reducing the number of parameters and calculations.

2) SPP MODULE

SPP (Spatial Pyramid Pooling) is one of the important measures for multi-scale pooling of high-level features in the target recognition algorithm to increase the receptive field. It can flexibly obtain the output with any available dimension by increasing the number of feature pyramid layers or changing the window size. The SPP module is able to improve the detection accuracy to a certain extent [55], [56]. Fig. 6 shows the structure of the SPP module. The SPP module owns the maximum pooling with a kernel size of 5 × 5, 9 × 9, 13 × 13, and a skip link [57]. The size of the maximum pooling kernel in the SPP module should be as close as possible or equal to the size of the input feature map. The SPP module realizes the fusion of local and global features, allows the neural network to extract features with different scales, and enriches the expressive ability of feature maps.

3) CROSS-LAYER BIDIRECTIONAL MODULE OF FEATURE FUSION

In a deep neural network, the deeper the number of layers is, the smaller the size of the feature map is, and the richer semantic information contained is. Due to having a larger resolution and retaining more spatial information of the original image, the shallow-layer feature map is conducive to determining the target position. The multi-scale feature network makes use of the advantages of different feature maps to realize the accurate detection of targets. In the proposed detection model, we add a scale of the feature map to perform multi-scale fusion and make full use of the shallow-layer spatial information to improve the accuracy of target locating. Fig. 7 presents the structure of the cross-layer bidirectional module of feature fusion.

The cross-layer bidirectional module of feature fusion has two kinds of connections from top to bottom and from bottom to top. Simultaneously, it has a horizontal connection from input nodes to output nodes to fuse more features. We add the deep-layer feature map to the previous feature map through upsampling. After an inverted residual block, we repeat the above operations until the obtained feature map fuses with a 104 × 104 feature map from the backbone. The shallow-layer feature information is transferred to the deep-layer feature maps through down-sampling to strengthen the feature pyramid. Furthermore, in the same size of feature maps, we add an extra edge to fuse more features without increasing the cost in the cross-layer bidirectional module of feature fusion. In this way, the detection model is capable of utilizing the shallow-layer information adequately and avoiding the loss of the deep-layer information to improve the locating accuracy and reduce the occurrence of false detection.

The use of the inverted residual block is to deepen the network and reduce the parameters as shown in Fig. 8(a). What is noteworthy is that we absorb the idea of residuals when performing down-sampling as shown in Fig. 8(b). The inverted residual block regards the Mish function as the activation function. The Mish function expression is:

$$Mish = x \times \tanh\left(\ln\left(1 + e^{x}\right)\right) \tag{1}$$

When the value of the Mish function is negative, the Mish function allows a relatively small negative gradient to flow in to ensure the flow of information. Compared to the performance of different activation functions in Squeeze Excite Net-18 for CIFAR 100 classification, the Mish activation function performs a more accurate detection [58]. Despite the computational complexity and time of the Mish function have increased a little, they are worth for the improvement of training stability and the improvement of final accuracy.

Different input features have different resolutions, and their contribution to output features is usually unequal. Hence, we introduce a simple fast attention mechanism to each input feature map. That is to say, we add the weight to achieve rapid normalization and fusion of the feature maps, and this process makes the network understand the importance of each input feature. The output feature map is:

$$P_{out} = \sum\nolimits_{i} \omega_i \Big/ \left(\varepsilon + \sum\nolimits_{i} \omega_i\right) \times P_i^{in} \tag{2}$$

where P_i^{in} is the different input feature map and ω_i is the normalized weight of the input feature map. The simple attention mechanism is equivalent to assigning different weights to each layer for fusion and allows the network to pay more attention to important layers.

C. YOLO HEAD

The YOLO head is in charge of prediction and classification of multi-scale targets. The proposed model also needs to add the corresponding the YOLO head due to adding a scale of the feature map. The sizes of the obtained YOLO heads are 104 × 104, 52 × 52, 26 × 26, and 13 × 13, respectively.

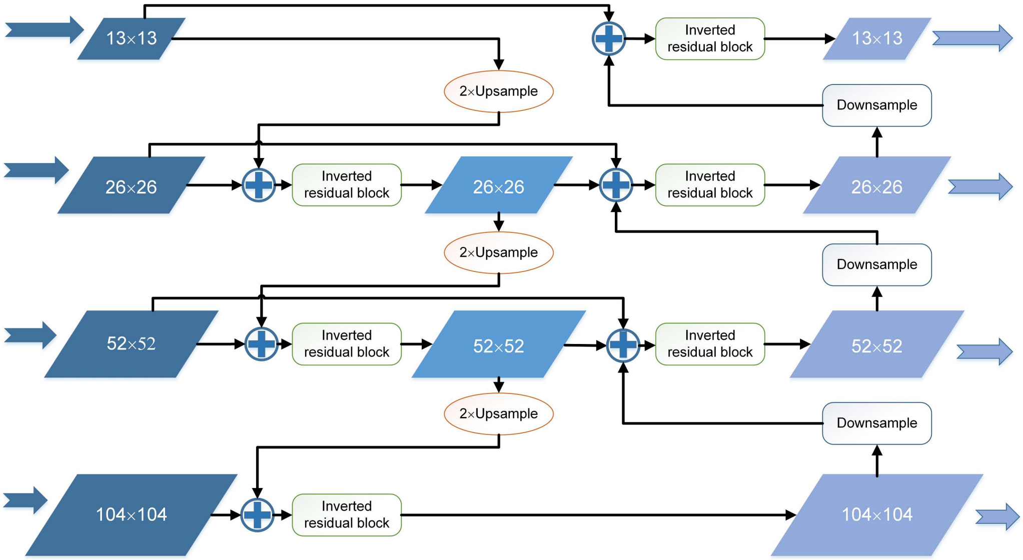


FIGURE 7. Structure of the cross-layer bidirectional module of feature fusion.

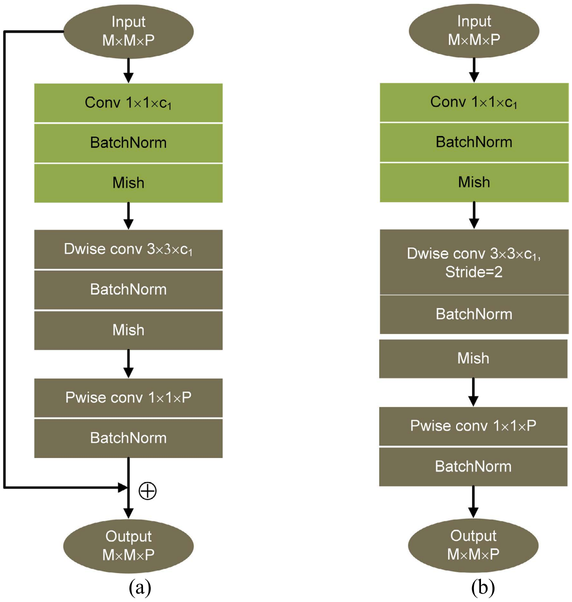


FIGURE 8. (a) Inverted residual block; (b) Block for down-sampling operation.

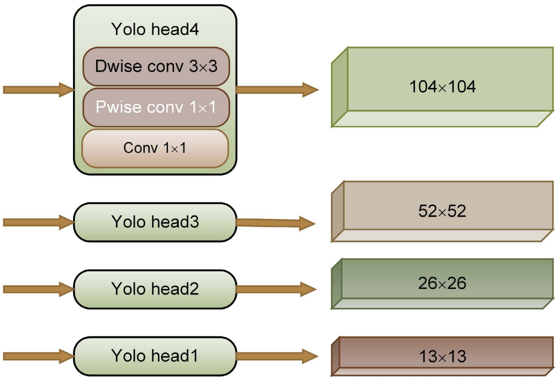


FIGURE 9. YOLO heads.

As shown in Fig. 9, we conduct the depthwise separable convolution that aims at increasing the dimensionality of the feature map and reducing the number of parameters in YOLO heads. At last, the YOLO head adjusts the dimensionality that the output needs by a 1 × 1 convolution layer.

D. ANCHOR BOX

The anchor box is proposed and applied in Faster R-CNN. The YOLOv2 takes the anchor box as a reference. Afterward, the various versions of YOLO series both take advantages of the anchor box. Different from the sliding window and RPN (Regional Proposal Network), the anchor box comes from the network. It can reduce the time cost and make the network model easier to learn. The YOLOv3 clusters anchor boxes by the k-means clustering algorithm and obtains three anchor boxes for each feature map.

The k-means clustering algorithm regards the distance between data points as a similarity index in the process of clustering iteration to find k classes in a given dataset. The center of each class is the mean point of all data points in the class. However, the random selection of the initial cluster center increases the randomness of the clustering [59].

Coupled with the interference of noise and external factors, the approach causes the uncertainty of classification, the mixing of different targets, and the same targets classified into different classes. Meanwhile, the diversity of targets in the complex dynamic traffic environment results in more samples to learning for the detection model. For this reason, the k-means clustering algorithm needs to adjust the sample classification and calculate the new cluster centers

TABLE 3. Detailed steps of the improved clustering algorithm.

```
Input: bounding box dataset $A$
Output: $c_k$, $k \in \{1,\ldots, K\}$
1: Randomly extract some data from the bounding box dataset $A$ to form a
   small batch $A_1$, $a_j \in A_1$, $j \in \{1,\ldots, 100\}$
2: Choose $a_j$ as seed point $c_1$ from $A_1$ at random
3: For $c_k$ ($k$<=K)
4:    For each bounding box $a_j$ from $A_1$
5:       Compute distance $d_j$ from the nearest seed point using (3)
6:       Add $d_j$ to get $Sum(d)$
7:    End for
8:    Choose a random value (0<$random$<1) and calculate $Random = Sum(d)$
      * $random$
9:    For $a_j$, $d_j$
10:       Calculate $Random = Random\text{-}d_j$
11:       If $Random$<=0
12:          Get the next seed point $c_k$
13:       End if
14:   End for
15:End for
16:For each small batch $A_m$ (m<=300) from the bounding box dataset $A$
17:    For $c_{k,m}$, $a_{j,m}$
18:       Calculate the shortest distance $d_{j,m}$ between $c_{k,m}$ and $a_{j,m}$ using (3)
19:    End for
20:    For $c_{k,m}$, $d_{j,m}$
21:       Recluster $c'_{k,m}$ using the median value of distances in the point group
22:    End for
23:    If $c'_{k,m} \neq c_{k,m}$
24:       go to step 17
25:    End if
26:End for
```

continuously. When there is a vast volume of samples in the dataset, the run-time overhead of the algorithm is expensive.

To solve these problems, we adopt the mini batch k-means++ clustering algorithm to cluster anchor boxes. The k-means ++ clustering algorithm is beneficial to initial cluster centers and effectively decreases the randomness of the clustering [60]. The mini batch k-means++ clustering algorithm that is an improved k-means algorithm reduces the run-time greatly as well as improves clustering accuracy as much as possible. The advantage of the mini batch method is that it does not employ all the data samples in the calculating process but extracts a part of the samples from different classes to represent their respective classes for clustering [61]. Due to the small number of samples for calculation, the mini batch method is capable of reducing the run-time accordingly [62].

The k-means clustering algorithm uses the Euclidean distance, but this method leads big bounding boxes to produce more errors than small bounding boxes. In the improved clustering algorithm, we introduce the IoU (Intersection over Union) to define the distance between the two bounding boxes. The distance between the two bounding boxes is:

$$d(box, seed) = 1 - IoU(box, seed) \tag{3}$$

where *IoU (box, seed)* is the ratio of the intersection to the union of the two boxes.

The detailed steps are shown as Table 3:

The clustering principle of the initial cluster centers is to make the distance between the cluster centers as far as possible. From step 8 to step 12 in Table 3, we take a random value and calculate the next "seed point" in the weight way. The implementation of this way is to take a random value *Random* that can fall in *Sum(d)* and then calculate *Random* = *Random-d* until *Random* <= 0. The bounding box at this moment is the next "seed point." Namely, when we take the value of *Sum(d)*random* where *random* is the weight and *Sum(d)*random=Random*, the value will fall into the interval of *d* with a high probability. The corresponding point is selected as the new "seed point" with a high probability.

In step 21, we set the cluster center as the point corresponding to the median value of distances in the point group to avoid the influence of noise points.

Actually, the process of clustering the cluster centers is one of function optimization. Assuming that given the number of classification groups K (K $\leq$ N and N is the number of data points), we divide the original data into K classes which are $S = \{S_1, S_2, \ldots, S_K\}$. The target function is optimized by:

$$\min J = \min \sum\nolimits_{n=1}^{N} \sum\nolimits_{k=1}^{K} r_{nk} \left\| d_n - \mu_k \right\|^2 \tag{4}$$

where μ_k represents the median value of the classification S_k. r_{nk} is 1 when the *nth* data point is classified into the *kth* cluster, otherwise, it is 0.

Differentiating μ_k with fixed r_{nk} and making the derivative equal to 0, we can search for the minimum J and get the value of μ_k. μ_k is:

$$\mu_k = \sum\nolimits_n r_{nk} d_n \Big/ \sum\nolimits_n r_{nk} \tag{5}$$

In the proposed detection model of the complex dynamic traffic environment for unmanned vehicles, we need to cluster twelve anchor boxes corresponding to four feature maps with different receptive fields.

E. LOSS FUNCTION

The purpose of choosing the various components of the loss function is to make coordinates, class, and confidence of predicted target achieve a good balance between the network output and the effect of target detection. As the basis for the deep neural network to judge the samples of false detection, the loss function dramatically affects the convergence effect of the neural network model [63]. The YOLOv3 model supports the form of the sum of squared errors (SSE) in the process of predicting position and regression of the bounding box, and it adopts the cross-entropy loss function in terms of confidence and class. The final total loss is in the form of the sum. Nevertheless, intuitively speaking, the center point of the bounding box is a certain relationship with the width and height. Therefore, we add an IOU loss in the loss function of the proposed detection model. Meanwhile, we apply BCE (binary cross-entropy) to the center offset loss of the predicted bounding box. If we divide the feature map into S × S grids and each grid generates B candidate boxes, we can obtain S × S × B bounding boxes ultimately. Composition and calculation of the loss function are:

$$L_{box} = -\sum\nolimits_{i=0}^{S^2} \sum\nolimits_{j=0}^{B} I_{i,j}^{obj} (2 - \hat{w}_i^j \times \hat{h}_i^j)[\hat{x}_i^j \log(x_i^j)$$

$$
\begin{aligned}
&+(1-\hat{x}_i^j)\log(1-x_i^j)] \\
&-\sum\nolimits_{i=0}^{S^2}\sum\nolimits_{j=0}^{B} I_{i,j}^{obj}(2-\hat{w}_i^j\times\hat{h}_i^j)[\hat{y}_i^j\log(y_i^j) \\
&+(1-\hat{y}_i^j)\log(1-y_i^j)] \\
&+\sum\nolimits_{i=0}^{S^2}\sum\nolimits_{j=0}^{B} I_{i,j}^{obj}(2-\hat{w}_i^j\times\hat{h}_i^j)[(w_i^j \\
&-\hat{w}_i^j)^2+(h_i^j-\hat{h}_i^j)^2]
\end{aligned} \tag{6}
$$

$$
\begin{aligned}
L_{conf} = &-\sum\nolimits_{i=0}^{S^2}\sum\nolimits_{j=0}^{B} I_{i,j}^{obj}[\hat{C}_i^j\log(C_i^j) \\
&+(1-\hat{C}_i^j)\log(1-C_i^j)] \\
&-\lambda_{noobj}\sum\nolimits_{i=0}^{S^2}\sum\nolimits_{j=0}^{B} I_{i,j}^{noobj}[\hat{C}_i^j\log(C_i^j) \\
&+(1-\hat{C}_i^j)\log(1-C_i^j)]
\end{aligned} \tag{7}
$$

$$
\begin{aligned}
L_{cls} = &-\sum\nolimits_{i=0}^{S^2} I_{i,j}^{obj}\sum\nolimits_{c\in class}[\hat{P}_i^j\log(P_i^j) \\
&+(1-\hat{P}_i^j)\log(1-P_i^j)]
\end{aligned} \tag{8}
$$

$$
L_{iou} = \sum\nolimits_{i=0}^{S^2}\sum\nolimits_{j=0}^{B} I_{i,j}^{obj} d_i^j \tag{9}
$$

$$
Loss = \lambda_{box}L_{box} + \lambda_{conf}L_{conf} + \lambda_{cls}L_{cls} + \lambda_{iou}L_{iou} \tag{10}
$$

where L_{box} is the regression locating loss of the predicted bounding box, L_{conf} is the confidence loss of the predicted bounding box, L_{cls} is the class loss of the predicted bounding box, and L_{iou} is the IoU loss of the predicted bounding box. $I_{i,j}^{obj}$ indicates whether the *jth* anchor box of the *ith* grid is responsible for this target. If the *jth* anchor box of the *ith* grid is responsible, $I_{i,j}^{obj} = 1$, otherwise $I_{i,j}^{obj} = 0$. $I_{i,j}^{noobj}$ indicates whether the *jth* anchor box of the *ith* grid is responsible for this target. If the *jth* anchor box of the *ith* grid is responsible, $I_{i,j}^{noobj} = 0$, otherwise $I_{i,j}^{noobj} = 1$. λ_{noobj} represents the weight of the confidence loss where the bounding box excludes the target. x_I^j, y_i^j, w_i^j, h_i^j, C_i^j, P_I^j, and d_i^j represent the center coordinates, the width, the height, the class probability, and the confidence of the *jth* bounding box of the *ith* grid, and distance between the ground truth box and the predicted bounding box, respectively. $\hat{x}_i^j$, $\hat{y}_i^j$, $\hat{w}_i^j$, $\hat{h}_i^j$, $\hat{C}_i^j$, and $\hat{P}_i^j$ represent the center coordinates, the width, the height, the class probability, and the confidence of the ground truth box, respectively. λ_{box}, λ_{conf}, λ_{cls}, and λ_{iou} represent the weight of L_{box}, L_{conf}, L_{cls}, and L_{iou}, respectively.

IV. EXPERIMENT AND DISCUSSION

Since the BDD100K dataset can reflect the complexity of the dynamic traffic environment more truly than other datasets and the KITTI dataset is commonly used in autonomous driving research, we choose the BDD100K dataset and the KITTI dataset for experiments. We verify the performance of the proposed detection model by comparing it with other advanced detection models. Training and deployment of models are performed using a server equipped with Intel Core i7-8700K CPU and NVIDIA GeForce GTX 1080Ti GPU card. All models are trained on two GPU cards. The validation experiments and the clustering experiments are performed in a personal laptop equipped with Intel Core i5-7300H CPU and NVIDIA GeForce GTX 1650 GPU card.

TABLE 4. Distribution of different classes in the BDD100K dataset.

Class	Bus	Light	Sign	Person	Bike
Number	16505	265906	343777	129262	10229
Class	Truck	Motor	Car	Train	Rider
Number	42963	4296	1021857	179	6461

TABLE 5. Distribution of different classes in the KITTI dataset.

Class	Pedestrian	Cyclist	Car
Number	4709	1627	33261

A. DATASET PROCESSING

The BDD100K dataset contains ten classes that are bus, light, sign, person, bike, truck, motor, car, train, and rider. It has 100000 images used for target detection. The images are divided into a training set of 70000, a test set of 20000, and a validation set of 10000. In the BDD100K dataset, the distribution of different classes is visible in Table 4.

The most number of the class is the car, and the fewest number of the class is the train of which the number is more than 5700 times different from that of the car. The second-to-last number of the class is the motor with a difference of more than 236 times from the number of the car. If the number of classes is extremely uneven, the neural network will differentiate the characteristics of the targets. For a large number of the class, the network will strength the ability to extract features. For a small number of the class, the network will weaken the ability to extract features. To avoid this case, we first remove labels of light, sign, and train that are not or uncommon dynamic targets of the complex traffic environment in the training set and validation set. Then, we merge the label information of the motor and the bike into the rider. Finally, we extract images in the processed dataset. When extracting images, we save all the images owning the bus and the rider and get a total of 14202 images in the training set and 1959 images in the validation set. Fig. 10 displays the distribution of classes in the processed dataset. The processed training dataset contains five classes: bus, car, person, rider, and truck. The class owning the most number of the class is the car with 142955, and the class owning the fewest number of the class is the truck with 7679. The number of them differs by 18 times, and the number of each class in the training set is greater than the empirical value of 2000.

The KITTI dataset has a total of 7481 images with labels. It contains eight classes, namely, car, vam, truck, pedestrain, person, cyclist, tram, and misc. In order to facilitate training, we merge the labels of vam, truck, and tram into the labels of car, merge the labels of person into the labels of pedestrain, and finally get three classes that are pedestrain, cyclist, and car, as shown in Table 5. The training set and the validation set are divided by 4:1.

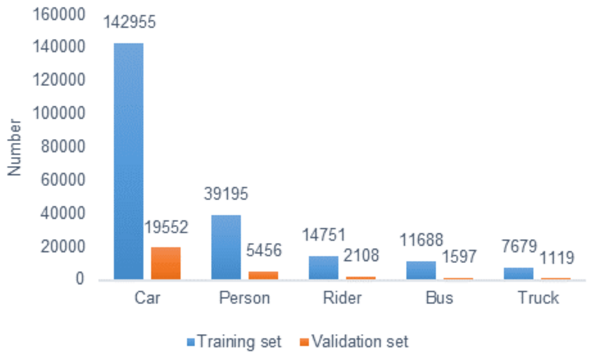


FIGURE 10. Distribution of classes in the processed dataset.

The training process is carried out for 120 epochs. The batch size is 8. The training learning rate is set by the cosine annealing algorithm, and the initial value is set to 0.001. We select the Adam optimizer to optimize the proposed detection model.

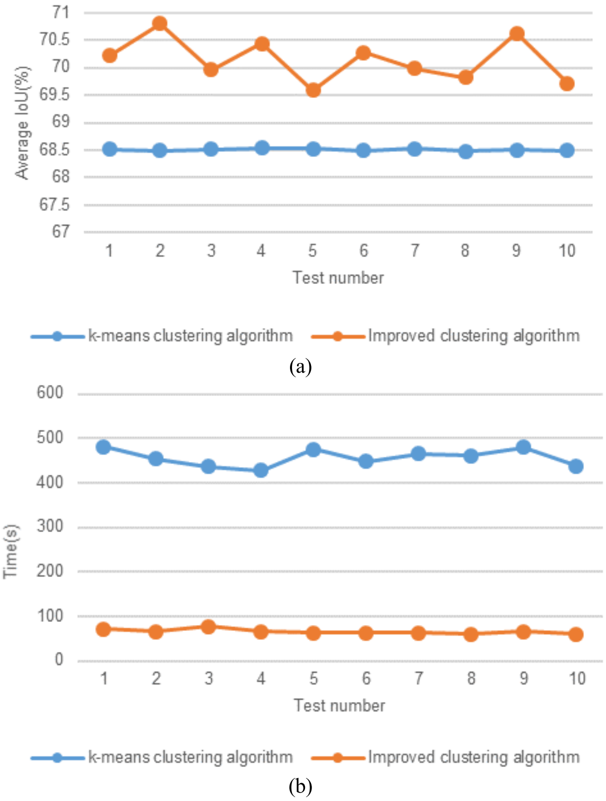
B. CLUSTERING OF ANCHOR BOXES

In order to cluster anchor boxes that conform to targets in the complex dynamic traffic environment, we perform clustering experiments in the datasets and compare the clustering effect of the improved clustering algorithm with that of the k-means clustering algorithm. We follow the routine of the original setting in YOLOv3, and we need to cluster twelve anchor boxes in the proposed detection model.

We employ the k-means clustering algorithm and the improved clustering algorithm to conduct ten experiments in the processed BDD dataset and record the run-time and the average IoU. As shown in Fig. 11(a), the average IoU obtained by using the k-means clustering algorithm stabilizes at 68.51%. In comparison, the average IoU obtained by using the improved clustering algorithm stabilizes at 70.15%, an increase of 2.39%. The improvement of average IOU also shows that the improved clustering algorithm reduces the influence of noise and interference from external factors due to using the medium value of distances in the point group. As shown in Fig. 11(b), it takes an average of 457.00 seconds to cluster the required anchor boxes by the k-means clustering algorithm, while it takes an average of 65.92 seconds to cluster the required anchor boxes by the improved clustering algorithm, a difference of 6 times.

In brief, the improved clustering algorithm enables to effectively reduce the impact of the randomness of the initial cluster center on the clustering effect and cluster anchor boxes that are more in line with the actual complex dynamic traffic environment. At the same time, the run-time decreases dramatically, which is very friendly to big data processing.

By the improved clustering algorithm, we get twelve anchor boxes in the processed BDD dataset: (4,7), (5,16), (6,27), (7,9), (9,16), (11,31), (15,20), (18,81), (23,35), (40,52), (68,103), and (130,198). In a similar way, we get twelve anchor boxes in the KITTI dataset: (5, 43), (9, 24), (12, 95), (13, 37), (18, 47), (25, 184), (26, 67), (28, 36), (42, 98), (43, 58), (68, 127), and (112, 209).



(a)

(b)

FIGURE 11. (a) Average IoU obtained by the k-means clustering algorithm and the improved clustering algorithm; (b) Run-time by the k-means clustering algorithm and the improved clustering algorithm.

C. PERFORMANCE COMPARISON WITH OTHER ADVANCED DETECTION MADELS

Commonly used evaluation indicators for evaluating the performance of the neural network include Precision(P), AP (Average Precision), Recall (R), F1-score (F1), and mAP (mean Average Precision). The calculation formulas of Precision, Recall, and F1 are respectively:

$$P = TP/(TP + FP) \tag{11}$$

$$R = TP/(TP + FN) \tag{12}$$

$$F1 = 2PR/(P + R) \tag{13}$$

where *TP* is the true positive sample, *FP* is the false positive sample, and *FN* is the false negative sample.

Precision refers to the proportion of true positive samples in all predicted positive samples. Recall refers to the proportion of the true positive samples in all true samples. Precision and Recall indicators are sometimes in contradictory situations, so we need to consider them comprehensively. F1 combines the results of Precision and Recall. The higher F1 is, the more effective the detection model of the complex dynamic traffic environment for unmanned vehicles is.

TABLE 6. (a) AP values and mAP values of various targets in the processed BDD dataset (IoU = 0.5); (b) F1 values of various targets and FPS in the processed BDD dataset (IoU = 0.5, confidence threshold = 0.5); (c) AP values, F1 values, mAP values of various targets and FPS about YOLOv5-l (IoU = 0.5, confidence threshold = 0.5).

(a)

Models	AP					mAP (%)
	Car (%)	Person (%)	Rider (%)	Bus (%)	Truck (%)	
Eifficientdet-D2[64]	42.67	13.62	14.05	37.16	2.70	22.04
YOLOv3+ MobileNetv3	60.31	29.06	33.69	53.82	31.96	41.77
YOLOv3+ MobileNext	61.31	32.04	34.79	54.42	30.45	42.60
YOLOv3[65]	66.10	40.07	44.07	62.19	40.80	50.64
YOLOv4[66]	66.24	41.98	40.44	59.07	32.67	48.08
Our model	73.59	53.07	47.38	65.04	45.42	56.90

(b)

Models	F1					FPS
	Car	Person	Rider	Bus	Truck	
Eifficientdet-D2	0.49	0.01	0.12	0.42	0.00	7.0
YOLOv3+ MobileNetv3	0.62	0.31	0.32	0.56	0.34	26.2
YOLOv3+ MobileNext	0.63	0.33	0.33	0.55	0.32	23.7
YOLOv3	0.68	0.43	0.46	0.62	0.44	12.8
YOLOv4	0.69	0.47	0.40	0.60	0.34	11.4
Our model	0.70	0.46	0.47	0.66	0.49	12.9

(c)

YOLOv5-l[67]	Car	Person	Rider	Bus	Truck
AP(%)	57.37	38.59	41.60	61.04	42.38
mAP(%)			48.20		
F1	0.60	0.31	0.40	0.62	0.47
FPS			16.2		

TABLE 7. AP values and mAP values of various targets in the KITTI dataset (IoU=0.5); (b) F1 values of various targets and FPS in the KITTI dataset (IoU=0.5, confidence threshold=0.5).

(a)

Models	AP			mAP(%)
	Car(%)	Cyclist(%)	Pedestrian(%)	
Eifficientdet-D2	61.97	11.13	20.52	31.21
YOLOv3+ MobileNetv3	91.13	57.63	55.33	68.04
YOLOv3+ MobileNext	91.44	59.35	56.00	68.93
YOLOv3	93.09	73.21	63.25	76.52
YOLOv4	92.10	68.22	63.20	74.51
Our model	93.96	83.78	76.58	84.77

(b)

Models	F1			FPS
	Car	Cyclist	Pedestrian	
Eifficientdet-D2	0.59	0.10	0.01	7.5
YOLOv3+ MobileNetv3	0.87	0.60	0.56	26.9
YOLOv3+ MobileNext	0.87	0.58	0.57	23.8
YOLOv3	0.90	0.73	0.63	13.5
YOLOv4	0.89	0.70	0.64	11.5
Our model	0.91	0.81	0.71	13.5

TABLE 8. Various model parameters in the detection models.

Models	Parameters	GFlops	Model size (M)
Eifficientdet-D2	8010499	8.93	31
YOLOv3+ Mobilenetv3	23204362	8.7	99
YOLOv3+ Mobilenext	22525418	9.1	93
YOLOv3	61545274	32.8	235
YOLOv4	64066926	30.0	244
YOLOv5-l	49848100	24.8	48
Our model	7901152	11.2	40

The mAP is able to evaluate the overall performance of the detection model. In the multi-target detection, the larger the AP value of each class is, the better performance of the detection model shows. The calculation formula is:

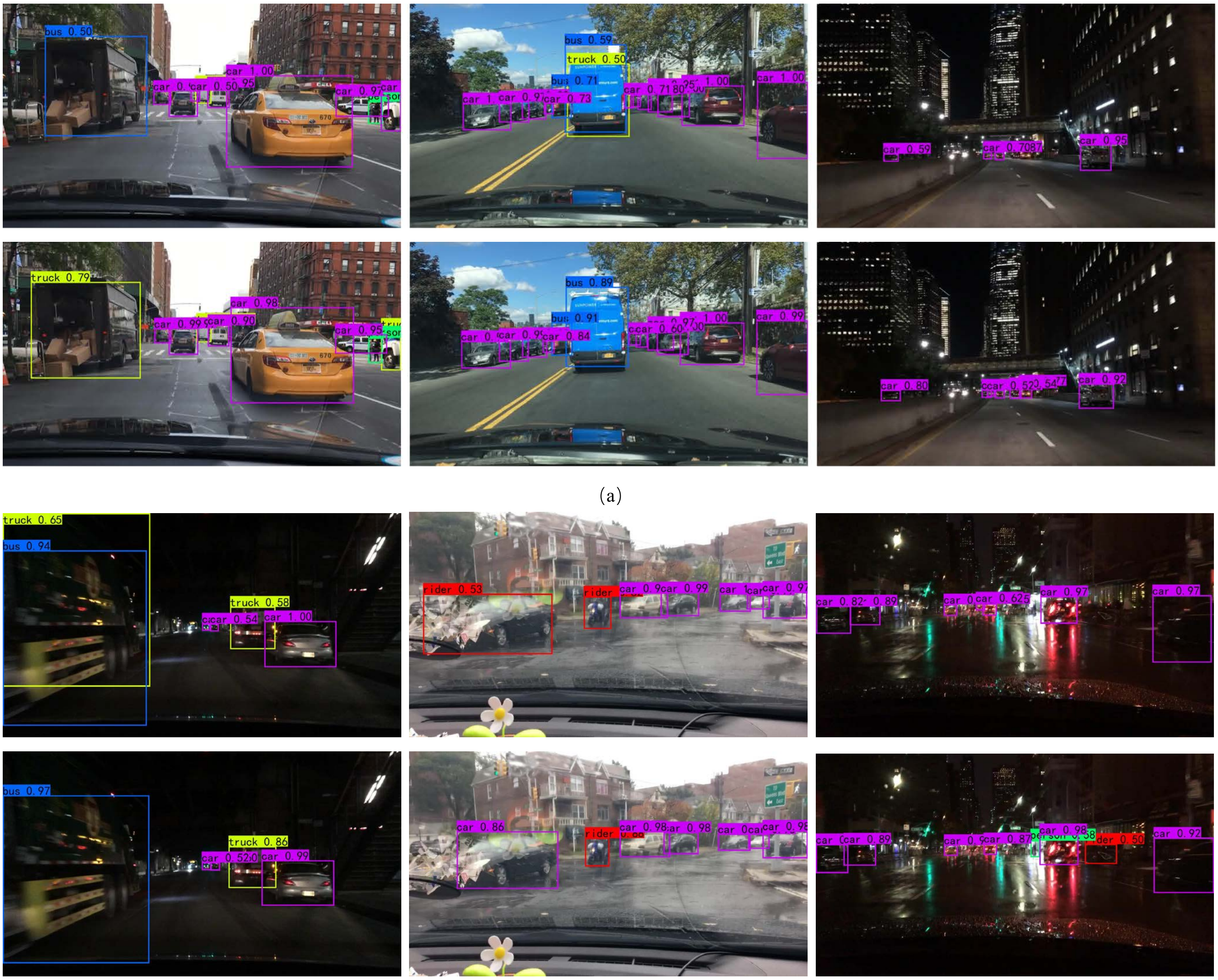
$$mAP = \sum\nolimits_{i=1}^{N} (AP)_i/N \tag{14}$$

where $(AP)_i$ is the AP value of each class, and N represents how many classes the dataset owns.

In order to demonstrate the advantages of the proposed detection model, we compare its performance with that of other advanced detection models in the experiments. The input images in other detection models are all 416 × 416 in size except for EfficientDet-B0. Table 6 and Table 7 show AP values, F1 values, mAP values of various targets and FPS in the processed BDD dataset and the KITTI dataset.

Considering the number of each class in the training set and the validation set, we can summarize that the size trend of the AP value of the class with a single color and shape is generally positively correlated with the number of the class in the proposed model. Nevertheless, the AP value of the person is inconsistent with the summary in the proposed BDD dataset. The number of the person in the validation set is 5456, ranking second, but the AP value is 53.07%, ranking third. This situation is related to the different postures, movements, and clothes of person. The situation of the pedestrian in the KITTI dataset is similar with that of the person.

In Table 6(a), the AP values in the proposed model are higher than those in other advanced detection models. The mAP value of the proposed model is 12.36% higher than that of YOLOv3, 18.34% higher than that of YOLOv4 in the processed BDD dataset. In Table 7(a), the mAP value of the proposed model increases respectively by 10.78% and 13.77% compared with that of YOLOv3 and YOLOv4 in the KITTI dataset. They mean that the proposed model has higher accuracy for detecting the complex dynamic traffic environment. Among the detection models, the F1 values of targets in the proposed model have a respectable performance as shown in Table 6(b) and Table 7(b). They illustrate that the proposed model is more effective for detecting the complex dynamic traffic environment in two datasets. For FPS, the proposed model is similar with YOLOv3. The FPS of the proposed model grows at 13.16% for that of YOLOv4 in the processed BDD dataset. In the KITTI dataset, the FPS of all the detection models improves slightly. The results show that the accuracy of the proposed detection model rises up significantly in two datasets by contrast, while the proposed detection model has no loss of detection speed. It indicates



(a)

(b)

FIGURE 12. (a)Performance of YOLOv3 and the proposed detection model;(b)Performance of YOLOv3 and the proposed detection model in rainy and at night. The first line is the performance of YOLOv3, and the second line is the performance of the proposed detection model in (a)and (b).

the proposed detection model is conducive to the safety of unmanned driving.

In Table 6(c), the evaluation indicators of YOLOv5-l are obtained by training and validation after further handling with the processed BDD dataset. We find YOLOv5-l is unable to complete the training on the processed BDD dataset. It illustrates that YOLOv5-l is unfriendly to detecting complex dynamic traffic environment for unmanned vehicles in the article. We filter out the bounding boxes of targets that is too small (the area ratio of the bounding boxes of targets to the image is less than 0.001) in the processed BDD dataset to obtain a new dataset. Nonetheless, the mAP value of the proposed model is 18.05% higher than that of YOLOv5-l. For the whole work process of unmanned driving, if the target detection is not accurate enough, the faster the detection speed is, the more dangerous the unmanned driving will be. Therefore, although YOLOv5-l has faster detection speed, it is still not suitable for the complex dynamic traffic environment in the article. For this reason, we no longer verify the performance of YOLOv5-l in the KITTI dataset.

Meanwhile, compared to MobileNext [52] as the backbone of YOLOv3, mobilenetv3 performs better on speed with a slight mAP loss.

Table 8 compares some model parameters of the proposed model and other advanced detection models. Flops is the total calculation amount of the model. The flops of the proposed model decrease by 65.85% for that of YOLOv3, 62.67% for that of YOLOv4, and 54.84% for that of YOLOv5-l. The weight size of the proposed model is 82.98% lower than that of YOLOv3, 83.61% lower than that of YOLOv4, and 16.67% lower than that of YOLOv5-l. In summary, the calculations, the number of parameters, and the model size of the

TABLE 9. AP values and mAP values of various targets in the processed BDD dataset.

Our model	AP										mAP	Changes
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes		
IoU=0.5	73.59%	——	53.07%	——	47.38%	——	65.04%	——	45.42%	——	56.90%	——
IoU=0.6	61.83%	15.98%	39.12%	26.29%	34.88%	26.38%	59.72%	8.18%	42.41%	6.63%	47.59%	16.36%
IoU=0.7	45.74%	26.02%	20.41%	47.83%	18.03%	48.31%	51.23%	14.22%	34.85%	17.83%	34.05%	28.45%

YOLOv3	AP										mAP	Changes
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes		
IoU=0.5	66.10%	——	40.07%	——	44.07%	——	62.19%	——	40.80%	——	50.64%	——
IoU=0.6	54.45%	17.62%	25.51%	36.34%	31.22%	29.16%	56.90%	8.51%	36.37%	10.86%	40.89%	19.25%
IoU=0.7	40.87%	24.94%	11.41%	55.27%	13.64%	56.31%	45.40%	20.21%	28.38%	21.97%	27.94%	31.67%

YOLOv4	AP										mAP	Changes
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes		
IoU=0.5	66.24%	——	41.98%	——	40.44%	——	59.07%	——	32.67%	——	48.08%	——
IoU=0.6	55.24%	16.61%	27.33%	34.90%	27.83%	31.18%	53.93%	8.70%	28.34%	13.25%	38.55%	19.82%
IoU=0.7	41.41%	25.04%	13.08%	52.14%	10.98%	60.55%	44.95%	16.65%	22.05%	22.19%	26.49%	31.28%

proposed model decrease dramatically in comparison, which is very beneficial to detect the complex dynamic environment for unmanned vehicles.

When IoU changes, the AP values, mAP values, and F1 values of the targets in the detection models change accordingly in Table 9 and Table 10. When IoU gradually rises, AP values, mAP values, and F1 values of the proposed models still maintain the top by contrast. Moreover, the average changes of AP values and the changes of mAP values of the proposed model are the lowest in the three models. The above data indicate that the predicted bounding boxes of the proposed model have higher confidence and more accurate locating. The lowest average changes of F1 values of the proposed model illustrate that the proposed model is more stable than YOLOv3 and YOLOv4 for detecting the complex dynamic traffic environment.

D. ABLATION EXPERIMENT AND VISUALIZATION

Table 11 lists the results of the ablation experiments in the processed BDD dataset. First of all, we replace the Darknet-53 with the MobileNetv3 in the backbone of YOLOv3, and we utilize the depthwise separable convolution in the detection model instead of the traditional convolution. We find that the number of parameters and calculations decreases notably in despite of the unsatisfactory accuracy. Secondly, we add the SPP module to the detection model. The mAP value rises up by 2.15 percentage points. Then, we adopt the proposed cross-layer bidirectional module of feature fusion and anchor boxes clustered by the improved clustering algorithm. The mAP value and the average F1 value improve significantly. Finally, we add the IoU loss to the loss function. The mAP value increases by 0.50 percentage points. The results in the ablation experiments show that the improvement methods according to the framework of YOLOv3 are effective.

In order to compare the performance of the proposed model and YOLOv3 more intuitively, we conduct a visual test as shown in Fig. 12. The YOLOv3 model regards the truck on the left as the bus in the first column of Fig. 12(a) and identifies the bus in the middle as the truck for the second time in the second column, while the proposed model correctly detects the target class in the complex dynamic traffic environment. Obviously, the proposed model detects the targets ignored by YOLOv3 in the third column of Fig. 12(a). Fig. 12(b) shows some special scenes on rainy days at night. YOLOv3 detects the truck on the left side of the first column, but there is no truck. In the second column, YOLOv3 identifies the car on the left as the rider on the rainy day. In the third column, YOLOv3 does not recognize the person and the rider due to the more complex traffic environment and blurry pictures.

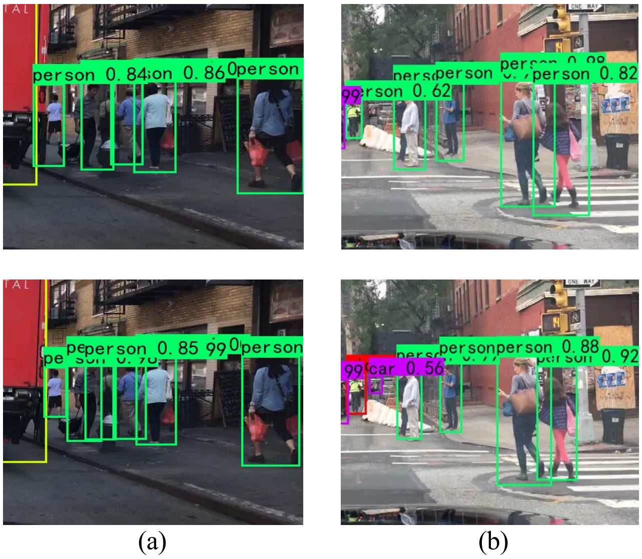


FIGURE 13. (a) Performance of YOLOv3 and the proposed detection model about the person label; (b) Performance of YOLOv3 and the proposed detection model about the person label and the rider label. The first line is the performance of YOLOv3, and the second line is the performance of the proposed detection model in (a)and (b).

TABLE 10. **F1 values of various targets in the processed BDD dataset.**

Our model	F1									
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes
IoU=0.5	0.70	——	0.46	——	0.47	——	0.66	——	0.49	——
IoU=0.6	0.65	7.14%	0.40	13.04%	0.40	14.89%	0.64	3.03%	0.48	2.04%
IoU=0.7	0.54	16.92%	0.30	25.00%	0.28	30.00%	0.59	7.81%	0.44	8.33%

YOLOv3	F1									
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes
IoU=0.5	0.68	——	0.43	——	0.46	——	0.62	——	0.44	——
IoU=0.6	0.60	11.76%	0.34	20.93%	0.38	17.39%	0.60	3.23%	0.42	4.55%
IoU=0.7	0.49	18.33%	0.22	35.29%	0.25	34.21%	0.54	10.00%	0.38	9.52%

YOLOv4	F1									
	Car	Changes	Person	Changes	Rider	Changes	Bus	Changes	Truck	Changes
IoU=0.5	0.69	——	0.47	——	0.40	——	0.60	——	0.34	——
IoU=0.6	0.61	11.59%	0.37	21.28%	0.33	17.50%	0.58	3.33%	0.33	2.94%
IoU=0.7	0.51	16.39%	0.25	32.43%	0.21	36.36%	0.53	8.62%	0.30	9.09%

TABLE 11. **Results of the ablation experiments.**

	Model	mAP (%)	Average F1	Param-eters	Gflops	Model size (M)
A	YOLOv3	50.64	0.53	62M	32.77	235
B	A+Backbone	41.77	0.43	23M	8.69	99
C	B+Backbone+ Depthwise separable convolution	38.58	0.40	68M	2.34	36
D	C+SPP module	40.73	0.42	86M	2.65	43
E	D+Cross-layer bidirectional module of feature fusion +Improved k-means clustering algorithm	56.40	0.56	79M	11.17	40
F	E+Loss function	56.90	0.56	79M	11.17	40

This is extremely detrimental to traffic safety. Experiments reveal that the proposed model can effectively avoid this kind of phenomenon. Fig. 13 shows performance of YOLOv3 and the proposed detection model about the person label and the rider label. YOLOv3 identifies five persons in Fig. 13(a), but there are six in the image. The proposed model can identify everyone. In Fig. 13(b), YOLOv3 detects the rider on the left as a person label, while the proposed model recognizes the rider with a red box correctly. Consequently, the proposed detection model is able to reduce the occurrence of false detections effectively and exhibits a better accuracy of target locating and a more superior performance compared with YOLOv3.

V. CONCLUSION AND FUTURE WORK

An excellent detection model of the complex dynamic traffic environment for unmanned vehicles can successfully realize the detection of the traffic environment for unmanned vehicles and improve the safety of unmanned driving. In order to balance the accuracy and speed of detecting the complex dynamic traffic environment for unmanned vehicles, we follow the framework idea of the YOLOv3 model and complete the following work:

1) To extract targets' features, we regard the MobileNetv3 that is on a foundation of inverted residual blocks with the SE structure as the backbone network, and the. depthwise separable convolution takes the place of the traditional convolution in the entire network model These works enable to reduce the number of parameters and calculations dramatically in the entire detection model.
2) In the enhanced feature fusion layers, the compress-and-expand module and the SPP module are used to continuously strengthen the feature fusion and reduce the number of parameters and calculations. In the cross-layer bidirectional module of feature fusion, we add a scale of the feature map to achieve multi-scale fusion of four feature maps, which avoids poor locating accuracy caused by insufficient use of shallow-layer information and the false detection caused by loss of deep-layer information.
3) Since there is a certain relationship between the center point and the width and height of the bounding box, we add an IoU loss according to the composition of the loss function of YOLOv3 and express the center offset loss of the predicted bounding box in the form of binary cross-entropy to realize the accurate regression of the detection model.
4) We improve the clustering algorithm and re-cluster anchor boxes by the improved clustering algorithm to avoid the randomness of selecting the initial clustering center and the influence of noise and interference from external factors. The improved clustering algorithm increases the clustering accuracy of bounding boxes as well as greatly reduces the run-time overhead of clustering.
5) According to the required detection targets in the complex dynamic traffic environment, we reprocess the

BDD100k dataset and the KITTI dataset. Subsequently, we perform the clustering experiments of anchor boxes and the comparison experiments among the proposed detection model and other advanced detection models.

The comparison results show that the number of parameters and calculations slows down dramatically, and the accuracy of the proposed detection model goes up significantly while the proposed detection model has no loss of detection speed. It means that the proposed detection model enables to improve the safety of unmanned driving significantly. Moreover, in contrast, the detection effect of the proposed model further improves through the visualization of the detection results, indicating the more superior performance of the proposed detection model of the complex dynamic traffic environment for unmanned vehicles.

In the next work, we will try to reduce the model size and the number of parameters and calculations by the pruning method, making the proposed detection model easier to deploy and more suitable for detecting the complex dynamic environment in the field of unmanned driving.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their corrections and helpful suggestions.

REFERENCES

[1] S. Yang, L. Gao, Y. Zhao, and X. Li, "Research on the quantitative evaluation of the traffic environment complexity for unmanned vehicles in urban roads," *IEEE Access*, vol. 9, pp. 23139–23152, 2021.

[2] Y. Zhao, Z. Li, L. Gao, and J. Xiong, "Road-feature-based multiparameter road complexity calculation model of off-road environment," *Math. Problems Eng.*, vol. 2018, Oct. 2018, Art. no. 1952792.

[3] Y.-N. Zhao, K.-W. Meng, and L. Gao, "The entropy-cost function evaluation method for unmanned ground vehicles," *Math. Problems Eng.*, vol. 2015, Oct. 2015, Art. no. 410796.

[4] H. Wang, L. Dai, Y. Cai, X. Sun, and L. Chen, "Salient object detection based on multi-scale contrast," *Neural Netw.*, vol. 101, pp. 47–56, May 2018.

[5] S. Shirmohammadi and A. Ferrero, "Camera as the instrument: The rising trend of vision based measurement," *IEEE Instrum. Meas. Mag.*, vol. 17, no. 3, pp. 41–47, Jun. 2014.

[6] X. Dai, "HybridNet: A fast vehicle detection system for autonomous driving," *Signal Process., Image Commun.*, vol. 70, pp. 79–88, Feb. 2019.

[7] A. Vishwakarma and M. K. Bhuyan, "Image fusion using adjustable non-subsampled shearlet transform," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 9, pp. 3367–3378, Sep. 2019.

[8] L. Wang, F. Chen, and H. Yin, "Detecting and tracking vehicles in traffic by unmanned aerial vehicles," *Automat. Construct.*, vol. 72, pp. 294–308, Dec. 2016, doi: 10.1016/j.autcon.2016.05.008.

[9] G. Mo and S. Zhang, "Vehicles detection in traffic flow," in *Proc. 6th Int. Conf. Natural Comput.*, Aug. 2010, pp. 751–754, doi: 10.1109/ICNC.2010.5583178.

[10] M. Dahl and S. Javadi, "Analytical modeling for a video-based vehicle speed measurement framework," *Sensors*, vol. 20, no. 1, p. 160, Dec. 2019, doi: 10.3390/s20010160.

[11] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *Int. J. Comput. Vis.*, vol. 73, no. 1, pp. 41–59, Jun. 2007.

[12] Y. Xu, G. Yu, Y. Wang, X. Wu, and Y. Ma, "A hybrid vehicle detection method based on viola-jones and HOG+SVM from UAV images," *Sensors*, vol. 16, no. 8, p. 1325, Aug. 2016, doi: 10.3390/s16081325.

[13] Z. Chen, C. Wang, C. Wen, X. Teng, Y. Chen, H. Guan, H. Luo, L. Cao, and J. Li, "Vehicle detection in high-resolution aerial images via sparse representation and superpixels," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 1, pp. 103–116, Jan. 2016, doi: 10.1109/TGRS.2015.2451002.

[14] Q. Li, L. Mou, Q. Xu, Y. Zhang, and X. X. Zhu, "R3-Net: A deep network for multioriented vehicle detection in aerial images and videos," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 5028–5042, Jul. 2019, doi: 10.1109/TGRS.2019.2895362.

[15] Z. Deng, H. Sun, S. Zhou, J. Zhao, and H. Zou, "Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 8, pp. 3652–3664, Aug. 2017, doi: 10.1109/JSTARS.2017.2694890.

[16] Y. Koga, H. Miyazaki, and R. Shibasaki, "A CNN-based method of vehicle detection from aerial images using hard example mining," *Remote Sens.*, vol. 10, no. 1, p. 124, Jan. 2018, doi: 10.3390/rs10010124.

[17] X. Liang, J. Zhang, L. Zhuo, Y. Li, and Q. Tian, "Small object detection in unmanned aerial vehicle images using feature fusion and scaling-based single shot detector with spatial context analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1758–1770, Jun. 2020, doi: 10.1109/TCSVT.2019.2905881.

[18] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[19] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[20] H. Wang, Y. Yu, Y. Cai, X. Chen, L. Chen, and Q. Liu, "A comparative study of state-of-the-art deep learning algorithms for vehicle detection," *IEEE Intell. Transp. Syst. Mag.*, vol. 11, no. 2, pp. 83–95, Mar. 2019.

[21] Y. Li, S. Li, H. Du, L. Chen, D. Zhang, and Y. Li, "YOLO-ACN: Focusing on small target and occluded object detection," *IEEE Access*, vol. 8, pp. 227288–227303, 2020.

[22] A. Marshall, "False positive: Self-driving cars and the agony of knowing what matters," in *WIRED*, May 2018. [Online]. Available: https://www.wired.com/story/self-driving-cars-uber-crash-false-positive-negative/

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012, doi: 10.1145/3065386.

[24] Z. Ren, Z. Xu, and E. Y. Lam, "End-to-end deep learning framework for digital holographic reconstruction," *Adv. Photon.*, vol. 1, no. 1, 2019, Art. no. 016004.

[25] Z. Ren, H. K.-H. So, and E. Y. Lam, "Fringe pattern improvement and super-resolution using deep learning in digital holography," *IEEE Trans. Ind. Informat.*, vol. 15, no. 11, pp. 6179–6186, Nov. 2019.

[26] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: A real-time object detection method for constrained environments," *IEEE Access*, vol. 8, pp. 1935–1944, 2020.

[27] 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.*, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[29] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[30] 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.

[31] J. Dai, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Conf. Neural Inf. Process. Syst.*, 2016, pp: 379-387.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[33] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2018, pp. 6154–6162.

[34] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 21–37.

[37] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

[38] 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.

[39] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[40] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[41] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[42] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The ApolloScape dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1067–1073.

[43] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 22–29.

[44] 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*.

[45] Y. Li, H. Wang, L. M. Dang, T. N. Nguyen, D. Han, A. Lee, I. Jang, and H. Moon, "A deep learning-based hybrid framework for object detection and recognition in autonomous driving," *IEEE Access*, vol. 8, pp. 194228–194239, 2020.

[46] X. Zhang, N. Li, and R. Zhang, "An improved lightweight network MobileNetv3 based YOLOv3 for pedestrian detection," in *Proc. IEEE Int. Conf. Consum. Electron. Comput. Eng. (ICCECE)*, Jan. 2021, pp. 114–118, doi: 10.1109/ICCECE51280.2021.9342416.

[47] W. Han, "A YOLOV3 system for garbage detection based on MobileNetV3_Lite as backbone," in *Proc. Int. Conf. Electron., Circuits Inf. Eng. (ECIE)*, Jan. 2021, pp. 254–258, doi: 10.1109/ECIE52353.2021.00061.

[48] S. Wang, J. Zhao, N. Ta, X. Zhao, M. Xiao, and H. Wei, "A real-time deep learning forest fire monitoring algorithm based on an improved pruned+KD model," *J. Real-Time Image Process.*, vol. 18, no. 6, pp. 2319–2329, May 2021.

[49] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[50] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[51] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 4510–4520.

[52] Z. Daquan, Q. Hou, Y. Chen, J. Feng, and S. Yan, "Rethinking bottleneck structure for efficient mobile network design," 2020, *arXiv:2007.02269*.

[53] K. H. Yang and Z. Y. Song, "Deep learning-based object detection improvement for fine-grained birds," *IEEE Access*, vol. 9, pp. 67901–67915, 2021.

[54] C. Nong, J. Zhang, Z. Liu, Q. Zeng, and T. Zhang, "Application of lightweight YOLOv4 in aircraft skin fault detection," in *Proc. 2nd Int. Conf. Comput. Vis., Image, Deep Learn.*, Oct. 2021, doi: 10.1117/12.2604633.

[55] 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.

[56] X. Wang, S. Wang, J. Cao, and Y. Wang, "Data-driven based tiny-YOLOv3 method for front vehicle detection inducing SPP-net," *IEEE Access*, vol. 8, pp. 110227–110236, 2020.

[57] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "PP-YOLO: An effective and efficient implementation of object detector," 2020, *arXiv:2007.12099*.

[58] D. Misra, "Mish: A self regularized non-monotonic activation function," 2019, *arXiv:1908.08681*.

[59] W. Min, X. Li, Q. Wang, Q. Zeng, and Y. Liao, "New approach to vehicle license plate location based on new model YOLO-L and plate pre-identification," *IET Image Process.*, vol. 13, no. 7, pp. 1041–1049, May 2019.

[60] W. He, Z. Huang, Z. Wei, C. Li, and B. Guo, "TF-YOLO: An improved incremental network for real-time object detection," *Appl. Sci.*, vol. 9, no. 16, p. 3225, Aug. 2019, doi: 10.3390/app9163225.

[61] D. Sculley, "Web-scale K-means clustering," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 1177–1178.

[62] Z. Ren, E. Y. Lam, and J. Zhao, "Real-time target detection in visual sensing environments using deep transfer learning and improved anchor box generation," *IEEE Access*, vol. 8, pp. 193512–193522, 2020.

[63] S. Lv, X. Cai, and R. Feng, "YOLOv3 network based on improved loss function," *Comput. Syst. Appl.*, vol. 28, no. 2, pp. 1–7, 2019.

[64] Bubbliiiing. *Eifficientdet*. Accessed: Nov. 2021. [Online]. Available: https://github.com/bubbliiiing/efficientdet-pytorch

[65] Bubbliiiing. *YOLOv3*. Accessed: May 2021. [Online]. Available: https://github.com/bubbliiiing/yolo3-pytorch

[66] Bubbliiiing. *YOLOv4*. Accessed: May 2021. [Online]. Available: https://github.com/bubbliiiing/yolov4-pytorch

[67] G. Jocher. *YOLOv5*. Accessed: Nov. 2021. [Online]. Available: https://github.com/ultralytics/yolov5

[68] J. Li, Y. Zhao, L. Gao, and F. Cui, "Compression of YOLOv3 via block-wise and channel-wise pruning for real-time and complicated autonomous driving environment sensing applications," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 6154–6162, doi: 10.1109/ICPR48806.2021.9412687.

SHIJUAN YANG received the B.S. and M.Sc. degrees from the Tianjin University of Science and Technology, Tianjin, China, in 2014 and 2017, respectively. She is currently pursuing the Ph.D. degree with the School of Mechanical Engineering, Beijing Institute of Technology, Beijing, China. Her research interests include test and evaluation, environment perception, and evaluation of environment complexity for intelligent vehicles or unmanned vehicles.

LI GAO received the B.S., M.Sc., and Ph.D. degrees in automobile transportation engineering from the School of Transportation, Jilin University of Technology, Changchun, China, in 1982, 1988, and 1996, respectively. He is currently a Professor with the School of Mechanical Engineering, Beijing Institute of Technology, Beijing, China. His research interests include transportation safety and logistics, environment perception, test and evaluation, in-vehicle information and intelligent transportation, and traffic behaviors.

YANAN ZHAO received the Ph.D. degree in systems engineering from the Northern Jiaotong University of Technology, in 2002. She did postdoctoral research in signal and information processing with the Beijing University of Posts and Telecommunication, Beijing, China, from 2002 to 2004. From June 2010 to June 2011, she was a Visiting Scholar with the University of Michigan Transportation Research Institute, University of Michigan, Ann Arbor, USA. She is currently an Associate Professor with the School of Mechanical Engineering, Beijing Institute of Technology, Beijing. Her research interests include test and evaluation, environment perception, evaluation of environment complexity for intelligent vehicles or unmanned vehicles, and intelligent transportation systems.