An Improved Lightweight Yolo-Fastest V2 for Engineering Vehicle Recognition Fusing Location Enhancement and Adaptive Label Assignment

Hairong Zhang, Dongsheng Xu, Dayu Cheng, Xiaoliang Meng, Geng Xu, Wei Liu, and Teng Wang

***Abstract*—Engineering vehicle recognition based on video surveillance is one of the key technologies to assist illegal land use monitoring. At present, the engineering vehicle recognition mainly adopts the traditional deep learning model with a large number of floating-point operations. So, it cannot be achieved in edge devices with limited computing power and storage in real time. In addition, some lightweight models have problems with inaccurate bounding box locating, low recognition rate, and unreasonable selection of positive training samples for the small object. To solve the problems, the article proposes an improved lightweight Yolo-Fastest V2 for engineering vehicle recognition fusing location enhancement and adaptive label assignment. The location-enhanced feature pyramid network structure combines deep and shallow feature maps to accurately localize bounding boxes. The grouping k-means clustering strategy and adaptive label assignment algorithm select an appropriate anchor for each object based on its shape and Intersection over Union. The study was conducted on Raspberry Pi 4B 2018 using two datasets and different models. Experiments show that our method achieves the optimal combination in speed and accuracy. Specifically, the mAP50 is increased by 7.02% with the speed of 11.24 fps under the engineering vehicle data obtained by video surveillance in a rural area of China.**

***Index Terms*—Adaptive label assignment, feature fusion, k-means clustering, lightweight model.**

Manuscript received 29 September 2022; revised 1 December 2022 and 7 February 2023; accepted 21 February 2023. Date of publication 27 February 2023; date of current version 9 March 2023. This work was supported in part by the Project of Major Scientific and Technological Achievements Transformation in Hebei Province of China (No. 22287401Z), in part by NSFC under Grant 41971352, in part by Alibaba Cloud Computing Co., Ltd., Funding (No. 17289315), and in part by Guangxi JMRH Funding (No. 202203). *(Corresponding author: Dayu Cheng.)*

Hairong Zhang is with the School of Environment and Spatial Informatics, China University of Mining and Technology, Xuzhou 221116, China, and also with the Key Laboratory of Natural Resources Monitoring in Tropical and Subtropical Area of South China, Ministry of Natural Resources, Guangzhou 510500, China (e-mail: hairong-zhang@cumt.edu.cn).

Dongsheng Xu and Geng Xu are with the School of Environment and Spatial Informatics, China University of Mining and Technology, Xuzhou 221116, China (e-mail: ts20160057a31@cumt.edu.cn; ts20160058a31@cumt.edu.cn).

Dayu Cheng is with the School of Mining and Geomatics Engineering, Hebei University of Engineering, Handan 056038, China (e-mail: chengdy@lreis.ac.cn).

Xiaoliang Meng is with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China, and also with the Key Laboratory of Natural Resources Monitoring in Tropical and Subtropical Area of South China, Ministry of Natural Resources, Guangzhou 510500, China (e-mail: xmeng@whu.edu.cn).

Wei Liu is with the School of Geography, Geomatics and Planning, Jiangsu Normal University, Xuzhou 221116, China (e-mail: liuw@jsnu.edu.cn).

Teng Wang is with the Lands and Resource Department of Guangdong Province, Surveying and Mapping Institute, Guangzhou 510500, China (e-mail: wangteng43@hotmail.com).

Digital Object Identifier 10.1109/JSTARS.2023.3249216

I. Introduction

With the development of economy, urban population, and transportation level, Chinese urbanization has entered a stage of rapid expansion. This will lead to dramatic changes in Chinese urban land use and increasingly prominent contradiction between people and land. In recent years, some illegal land use behaviors have occurred frequently, especially the illegal construction of cultivated land, forest land, and river. In addition, coupled with the serious national conditions of land pollution and soil erosion, we must cherish land resources and strengthen the fight against illegal land use. Engineering vehicles (such as trucks and excavators) are important tools for illegal land use. Therefore, identifying engineering vehicles from video is a useful way to monitor illegal land use. The traditional method identifies engineering vehicles manually through surveillance video by staff [5], a time-consuming and labor-intensive process. With the rapid advancement of deep learning, automatic image recognition based on the convolutional neural networks (CNN) has become widespread in traffic management [6], [7], [8], [9], crime prevention [10], [11], [12], anomaly detection [13], [14], [15], [16], fire warning [17], human detection [18], [19], [20], [21], [22], and agricultural management [23], among other fields. This also enables the identification of engineering vehicles from video in real time. However, the real-time video captured by the online camera must be uploaded to a cloud or local server on which the deep learning recognition model has been deployed [24], [25]. This process is affected by network bandwidth, preventing data transmission in real time. Not only are a significant number of operations concentrated on the server but the server's performance requirements are also relatively high. Consequently, in the process of video recognition, deploying computing power at the front end is a crucial way to alleviate the pressure of data transmission and central server computing. Some researchers have proposed smart devices that advance computing tasks to the network edge with the advent of edge computing [26], [27]. The object recognition model is directly deployed in the edge device, and the edge device's computing resources are used to perform a portion of the computing tasks required for object recognition [28], [29], thereby enhancing real-time recognition and relieving the burden on the central server.

However, because current mainstream deep learning object recognition models require large floating-point operations (FLOPs) [30], [31], [32], real-time recognition cannot be achieved in edge devices with limited computing and storage resources. Therefore, developing a lightweight model for object recognition in land and resource management has become a popular research direction [33], [34], [35].

Currently, some researchers are investigating lightweight object recognition models. Nikouei et al. [36] developed a lightweight human detection model L-CNN using depthwise separable convolutions and only 23 layers. The edge device Raspberry PI 3 Model B has an average recognition speed of 1.79 frames per second (FPS) for 224 × 224 video images. Even though this model has a small number of parameters, the computation required is still substantial, and real-time detection cannot be achieved on devices with limited computing resources. Steinmann et al. [37] replaced the SSD detector's backbone network with a lightweight backbone network PELEE and combined the filter pruning strategy to reduce the number of parameters. On Jetson AGX, the recognition speed is 24 times faster than the original model, but the recognition accuracy suffers. Based on YOLO-Lite as the backbone network, Zhao et al. [38] added a residual block, balanced the high- and low-resolution subnetworks, and proposed the mixed YOLO V3-Lite lightweight model. Using Jetson AGX Xavier equipment, the speed of recognizing 224 × 224 video images can reach 43 FPS. Still, the number of parameters is substantial, reaching 20.5 MB. Balamuralidhar et al. [28] introduced the MultEYE network in the UAV traffic detection system MultEYE. The network implements pruning to reduce the number of CSP Bottleneck and convolution blocks and adds a space-to-depth layer to decrease the space size and increase the number of channels. The speed of recognizing 512 × 320 video images on NVIDIA Xavier NX can reach 29 FPS. The head, neck, and backbone of the lightweight model of RangiLyu [39] proposed an anchor-free Nanodet model, which can recognize data with a size of 320 × 320 on a mobile ARM CPU at a speed of 97 FPS. However, the anchor-free model has significant issues with positive and negative sample imbalance and semantic ambiguity, leading to the phenomenon of missing objects. Amudhan and Sudheer [40] proposed a lightweight model PmA that can detect small objects in remote sensing images effectively and quickly by extracting more feature information from shallow features and transferring shallow features to deep features for detection. Compared with YOLO V3-tiny and YOLO V4-tiny, Jetson Nano's detection speed is 32% faster. Liu et al. [41] addressed the issues of limited UAV computing resources and too small objects by enhancing the network structure and target prediction frame selection algorithm of YOLO V4. This enhanced the recognition accuracy. Achieving 23.8 FPS on Nvidia Jetson TX2 and 9.6 FPS on Raspberry Pi 4B. Dog-qiuqiu [42] proposed the Yolo-FastestV2 model for detection in real time on mobile devices. The model is intended to replace the backbone network of YOLO V5 with ShuffleNet V2 while simultaneously lightening the feature pyramid network (FPN) structure; the parameter size of the model is just 237.55 kB. On the Mate 30 Kirin 990 CPU, the image recognition speed of 352 × 352 can reach more than 300 FPS. The real time detection effect can also be achieved on embedded devices with limited computing resources. However, Yolo-Fastest V2 lacks rich feature fusion, which prevents it from making full use of the rich location information in shallow feature maps and from selecting suitable positive training samples effectively. This results in insufficient extraction of location and semantic features of small objects, which leads to inaccurate positioning of the bounding boxes (bboxes) of small objects and a low recognition rate.

Yolo-FastestV2 is a widely used lightweight object recognition model. This article proposes a method of fusing location-enhanced FPN and adaptive label assignment to address the issues encountered by Yolo-FastestV2. This method improves the FPN structure by fusing feature maps of different scales to ensure that the object's location and feature information are fully extracted. A grouping k-means clustering strategy and an adaptive label assignment algorithm are proposed to assign appropriate anchor boxes to each object to combat the issues of low recognition rate and unreasonable selection of positive training samples. The algorithm assigns each object in the image an anchor with a similar shape and the largest Intersection over Union (IoU). In other words, a suitable positive training sample is selected to facilitate the regression of the bbox.

In summary, the following are the article's contributions.

1) Feature fusion is added based on the original model FPN. Integrate the shallow feature map rich in location information with the deep feature map channel rich in semantic information to enhance the localization ability of the model.
2) Introducing a threshold α and proposing a grouping k-means clustering strategy. According to the adjustment of the α value, multiscale anchors are generated for the small object. So, the small-scale anchor box can be set reasonably to make the small bbox easier to learn.
3) An adaptive label assignment algorithm based on shape and IoU is proposed. Based on the shape label assignment, an IoU label assignment algorithm is introduced. The shape threshold and IoU threshold are dynamically calculated by calculating the mean and variance of IoU and the mean and variance of the ratio of the shape. Under the dual constraints of shape and IoU, an appropriate anchor box is assigned to each object.

II. Proposed Method

As shown in Fig. 1, the model consists of three components: the backbone network ShuffleNet V2, the FPN, and the prediction head. This article improves the Yolo-FastestV2 model in three aspects: FPN structure, anchor box settings, and label assignment.

In recent years, ShuffleNet V2 has been an excellent network for lightweight feature extraction. It consists of three stacked Shufflev2Block convolution blocks. The entire network ensures optimal performance for feature extraction by minimizing memory access. The location-enhanced FPN structure (red dotted-line) will integrate the location information of the shallow features into the deep features to improve the model's

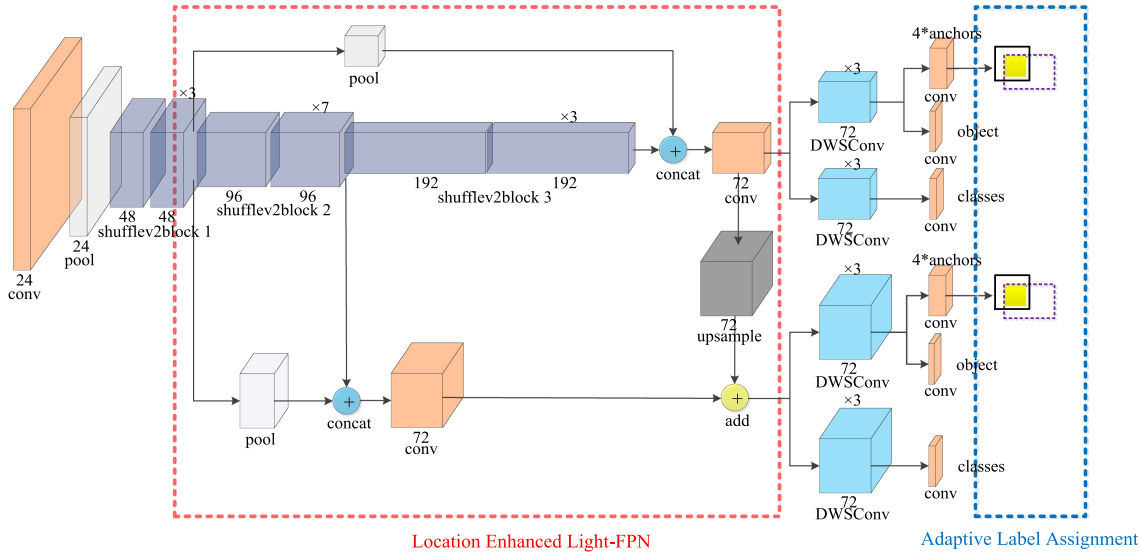

Fig. 1. Overall framework of the improved Yolo-Fastest V2.

ability to locate the bbox of a small object. In the label assignment algorithm of the prediction header (blue dotted-line), the grouping k-means clustering strategy generates reasonable anchors for small objects, while the adaptive label assignment algorithm matches appropriate anchors for each object. Details are as follows.

A. *Location-Enhanced FPN*

In Fig. 2(a), the original FPN structure predicts small objects using the feature map obtained by convolution of the third ShuffleV2Block convolution block in the ShuffleNet V2 network structure, in conjunction with 1 × 1 convolution. Then, the feature map obtained by the 1 × 1 convolution is upsampled. The feature map obtained by the second ShuffleV2Block convolution block is subjected to concatenation feature fusion. Finally, the 1 × 1 convolution is performed to predict the object whose scale is greater than that of the small object. Thus, FPN utilizes only two layers of shallow feature maps, lacks rich object location information, and cannot fully extract semantic information. This will result in inaccurate center coordinate positioning when the model predicts a bbox of small object.

This article proposes a location-enhanced FPN structure [see Fig. 3(b)], which primarily contributes to feature fusion by introducing 1/8 times the shallow feature map obtained by the first ShuffleV2Block convolution block into the deep feature map.

For medium- and large-scale objects, the feature map obtained from the first ShuffleV2Block convolution block is downsampled by a factor of 4 to obtain C1. Second, obtain C11 by performing concatenation feature fusion on C1 and the feature map obtained by the third ShuffleV2Block convolution block. C11 is subjected to 1 × 1 convolution to predict objects of medium and large scale.

For small-scale objects, the result after C11 convolution is first upsampled by two times to obtain C13. Second, C2 is obtained by downsampling by a factor of 2 the feature map obtained by the first ShuffleV2Block convolution block. The feature map obtained by C2 and the second ShuffleV2Block convolution block is fused by concatenation to obtain C21. C21 incorporates deep semantic information. C22 is obtained after 1 × 1 convolution. After each concatenation feature fusion operation, 1 × 1 convolution is used because during the whole model training process, the convolution operation can extract the effective feature information of the previous feature map and

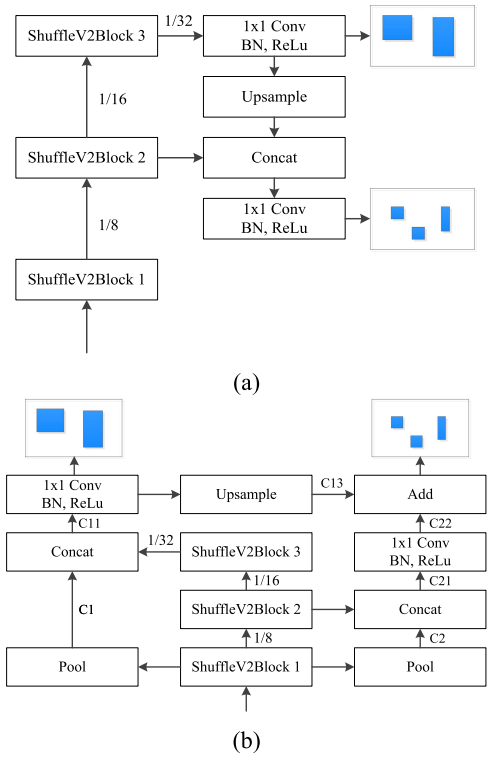

Fig. 2. (a) FPN. (b) Location-enhanced FPN.

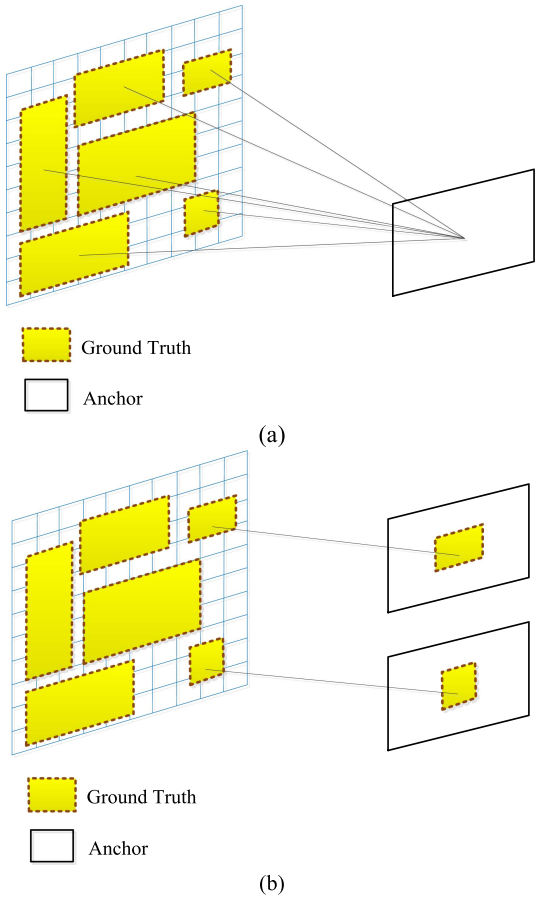

Fig. 3. (a) Small objects are aggregated as a whole into a class of anchors. (b) Smaller objects do not match the anchors obtained by clustering.

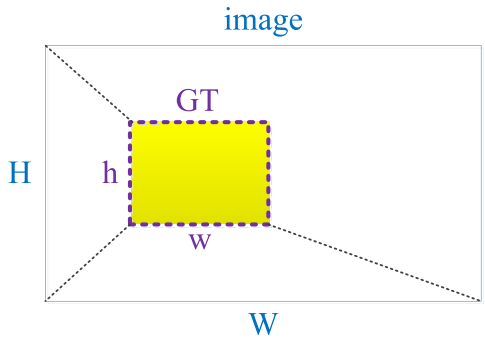

Fig. 4. Ratio of width to height.

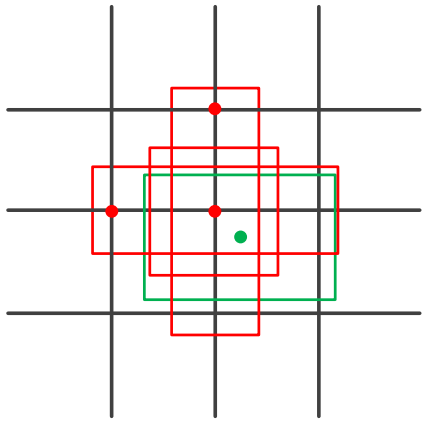

Fig. 5. Shape assignment (Yolo-Fastest V2).

weaken the noise information. Finally, C22 and C13 perform addition feature fusion to predict small objects. Through the method of addition feature fusion, shallow and deep features are fully fused, resulting in a fused feature map with rich object location information that improves the original model's ability to locate bboxes. This structure enables the prediction of large and small objects to obtain the fusion of shallow features with rich location information, thereby improving the locating accuracy.

B. Grouping k-Means Clustering Strategy

For anchors-based object recognition, appropriate anchors must be set before label assignment. However, the object's size in the image is either large or small. If we directly apply k-means clustering to the bboxes of the training set, it is simple to cluster the bboxes of some very small objects into a wide range of anchors [see Fig. 3(a)]. This is due to the influence of threshold k in the k-means method, which results in assigning such anchors to small objects [see Fig. 3(b)]. Ultimately, the anchors cause harmful gradient values to be passed back.

This article proposes a grouping k-means clustering strategy in response to the issue. As shown in Fig. 4, this strategy includes a threshold α. Then, compare the width and height of the object's ground-truth (GT) with the image's width and height. The GT whose minimum value of this ratio is less than the threshold α is clustered into $n(n\geq 1)$ classes; GT boxes larger than α are clustered into $m(m\geq 1)$ classes

$$\text{Number of anchor box types} = \begin{cases} n, & \min(h/H,\ w/W) < \alpha \\ m, & \min(h/H,\ w/W) \geq \alpha. \end{cases}$$

The threshold α is used as the demarcation point between the small object and the medium object. It can adjust the threshold α by analyzing the relative width and height distribution of the dataset object so that the small objects form a single set. The k-means clustering is performed on this set to obtain three anchors of different scales, allowing each small object to be assigned a more reasonable label. For the setting of m and n values, models from the Yolo series [26] usually cluster into three types of anchors for various object levels. Therefore, this article continues the Yolo method and sets n and m to 3, respectively.

C. Adaptive Label Assignment

After acquiring appropriate anchors, combine shape label assignment (see Fig. 5). The object is matched with the appropriate anchor, which is then assigned to cls and reg labels. However, the original method of shape label assignment causes two issues: First, the fixed shape threshold cannot guarantee that each object matches an anchor. In other words, the selection of positive training samples is unreasonable. For example, objects larger than the shape label assignment threshold at different levels [see Fig. 6(a) and (b)]; second, it is possible to match multiple anchors with similar shapes, among which the anchors with

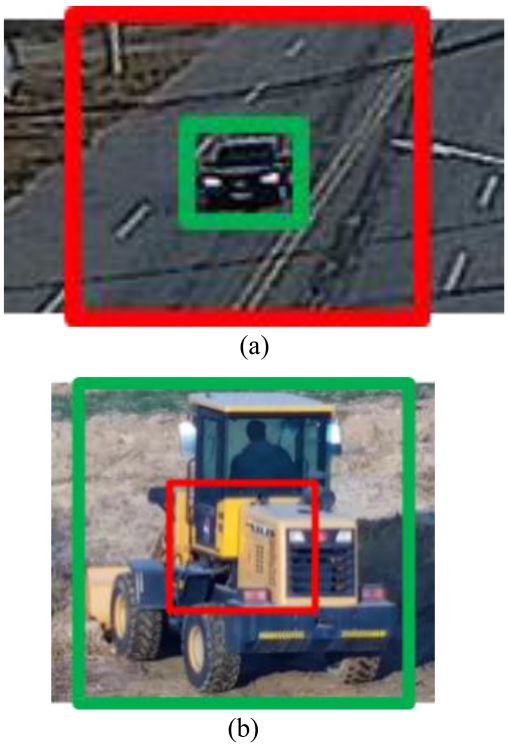

Fig. 6. (a) GT is too small. (b) GT is too large.

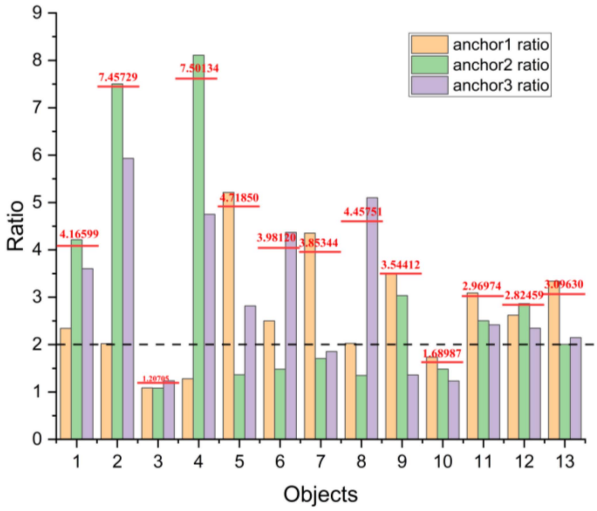

Fig. 7. Adaptive shape similarity threshold constraint.

low IoU participate in the backpropagation, which increases the complexity of model training.

This article proposes an adaptive label assignment strategy inspired by the ATSS algorithm [43]. The strategy is divided into shape similarity and maximum IoU. In terms of shape, first, at the same level, calculate the mean and standard deviation of the aspect ratio (r_w, r_H) of each GT and anchors. Calculating the sum of the mean and standard deviation yields the upper boundary constraint value for each object's various anchors. As shown in Fig. 7, this constraint guarantees that each object is assigned to at least one similarly shaped anchor. The shape similarity threshold is calculated as follows:

$$d_W^{ij} = \begin{cases} \frac{W_{gt}^i}{W_{\text{anchor}}^j}, & \text{if } r_w \geq 1 \\ \frac{W_{\text{anchor}}^j}{W_{gt}^i}, & \text{if } r_w < 1 \end{cases}, \quad d_H^{ij} = \begin{cases} \frac{H_{gt}^i}{H_{\text{anchor}}^j}, & \text{if } r_H \geq 1 \\ \frac{H_{\text{anchor}}^j}{H_{gt}^i}, & \text{if } r_H < 1 \end{cases} \tag{1}$$

$$r_{ij} = \max\left(d_W^{ij}, d_H^{ij}\right) \tag{2}$$

$$\mu_i = \frac{\sum_{j=1}^{m} r_{ij}}{m} \tag{3}$$

$$\sigma_i = \sqrt{\frac{\sum_{j=1}^{m} \left(\mu_i - r_{ij}\right)^2}{m}} \tag{4}$$

$$\alpha_{\text{shape}}^i = \mu_i + \sigma_i \tag{5}$$

in (1)–(5), H_{gt}^i represents the height of the ith object GT box; H_{anchor}^j represents the height of the jth anchor; W_{gt}^i denotes the width of the ith object GT box; W_{anchor}^j represents the width of the jth anchor; d_W^{ij} denotes the ratio of the width of the jth anchor of the ith object GT box; d_H^{ij} represents the ratio of the height of the jth anchor of the ith object GT box; m represents the number of anchors; r_{ij} denotes the maximum value of the ratio of the width and height of the jth anchor of the ith object GT box; μ_i represents the r-mean of the ith object GT box; σ_i denotes the r-variance of the ith object GT box; α_{shape}^i represents the shape threshold of the ith GT box.

In terms of IoU, determine the mean and standard deviation of the IoU for each object's GT box and anchors at the same level. The sum of the mean and standard deviation is used as the lower boundary constraint value of each object's GT box's IoU. This constraint combines the results of the shape similarity constraint to determine the participating positive sample anchors in training. If the anchors assigned based on shape similarity do not meet the adaptive IoU threshold, they are included in the label assignment for the following level. This ensures that each object has a corresponding anchor. The lower bound threshold is calculated as follows:

$$\mu_{\text{IoU}}^i = \frac{\sum_{i=1}^{m} \text{IoU}_{ij}}{m} \tag{6}$$

$$\sigma_{\text{IoU}}^i = \sqrt{\frac{\sum_{j=1}^{m} \left(\mu_{\text{IoU}}^i - \text{IoU}_{ij}\right)^2}{m}} \tag{7}$$

$$\alpha_{\text{IoU}}^i = \mu_{\text{IoU}}^i + \sigma_{\text{IoU}}^i \tag{8}$$

in (6)–(8), IoU_{ij} represents the IoU of the ith object GT box and the jth anchor; μ_{IoU}^i denotes the mean of all IoUs for the ith object GT box; σ_{IoU}^i represents the standard deviation of all IoUs for the ith object GT box; α_{IoU}^i denotes the IoU threshold for the ith object GT box.

In a lightweight network structure, too many layers of pyramid structure will affect the recognition speed, which is not conducive to embedded deployment and operation. The proposed method adapts anchors at different scales for each object at the same feature scale level. Therefore, the method is also applicable to network structures without FPN. The proposed method is at the multiscale anchor level based on the relationship between

TABLE I
MODEL TRAINING ENVIRONMENT

Software and Hardware	Parameter
Operating System	Windows10
GPU	NVIDIA GeForce RTX 2080 Ti 8 GB
cuDNN	V7.6.5 for CUDA10.2
IDE	Pycharm
Programming Language	Python
Frame	Pytorch

TABLE II
EMBEDDED DEVICE OPERATING ENVIRONMENT

Software and Hardware	Parameter
Device	Raspberry Pi 4B 2018
Operating System	Linux 32bit
CPU	arm v7l
Programming Language	C++
Frame	NCNN

NCNN: a high-performance neural network inference computing framework produced by Tencent.

each object's GT box and anchors. The adaptive label assignment strategy will dynamically adjust the shape similarity threshold and IoU threshold so that the object matches the anchor in terms of shape similarity and overlap. It can mitigate the issue of unreasonable positive training samples selection to some extent.

III. EXPERIMENTAL RESULTS AND ANALYSIS

This article uses the engineering vehicle image collected by the monitor as the data source. First, train the model. Then, ablation experiments are conducted to determine the effect of the location-enhanced FPN, the grouping k-means clustering strategy, and the adaptive label assignment on the model. Finally, it is compared with other mainstream-lightweight object detection models to validate the method's advancement.

A. Data Introduction and Experimental Environment

There are 679 images with 1080∗1920 pixels in the dataset of this experiment. Divide the dataset into training set and testing set by 8:2. There are 543 training data and 136 testing data. The recognized object types include car, truck, agricultural car, and excavator. In training set, the number of vehicles is 587, 389, 161, and 131, respectively. In testing set, the number of vehicles is 166, 110, 36, and 37, respectively. During the training phase, transfer learning and data augmentation (e.g., rotations, gamut transformation, etc.) are used to train the model.

1) Description of the Experimental Environment: The model was trained on an NVIDIA GeForce RTX 2080 Ti GPU with 8 GB of memory (see Table I). As shown in Fig. 8, a Raspberry Pi 4B 2018 with an ARM v7l CPU is used as a limited-resource edge computing device to infer the model (see Table II).

Fig. 8. Raspberry Pi 4B 2018.

2) Model Training: Due to privacy restrictions on surveillance video, only a limited number of images can be obtained. Therefore, this article only divides the train set and the test set, not the val set, to ensure a sufficient number of train sets. The improved model is pretrained on MS COCO 2017s train set to obtain pretrained weights. Other methods employ official COCO pretrained weights. On the premise that all models do not freeze the parameters of any layer, the pretrained model is fine-tuned using transfer learning on the train set in this article. The experiment adopts data augmentation based on image processing because it can avoid overfitting of the model on the dataset with a small amount of data. It has a weak impact on the recognition accuracy.

3) Model Parameters: The minibatch size is set to 4 for all other methods, and all other parameters retain their official configuration. The improved model is trained with an initial learning rate of 0.001 for 240 epochs. At the 130th, 160th, 178th, and 185th epochs, the learning rate decreases by a factor of 10. Minibatch size is set to 4, and the stochastic gradient descent algorithm is implemented. In this experiment, CrossEntropyLoss and CIoU Loss are used as loss functions for classification and regression. The before and after background classification loss $\mathcal{L}_{\text{obj}}$ has a weighting factor of 34. The type classification $\mathcal{L}_{\text{cls}}$ weighting factor is 64. The bbox regression $\mathcal{L}_{\text{reg}}$ weighting factor is 3.4. In the inference phase of all methods, the nonmaximum suppression threshold is 0.45, the classification (cls) threshold is 0.5, the object (obj) threshold is 0.5, and the confidence (conf) scores on the bbox is the cls threshold multiplied by the obj threshold. The threshold α is determined primarily by the distribution of the data in two steps. First, the determination of the threshold α requires an analysis of the relative width and height distribution of the bounding box for all objects in the dataset. Second, α value is determined according to the distribution of the relative width and height of the small object. In Fig. 9, the relative width and height distributions of small objects in the training set are within 0.1 at the smallest increment. Therefore, the threshold α is set to 0.1 in grouping k-means clustering.

TABLE III
VALIDATION OF THE PROPOSED METHOD ON YOLO-FASTESTV2 (MONITORING DATASET)

Method	Size	FLOPs	mAP50
Yolo-Fastest V2	352*352	0.11G	42.90
Yolo-Fastest V2+LE-FPN	352*352	0.11G	43.91(+1.01)
Yolo-Fastest V2+GKC	352*352	0.11G	44.94(+2.04)
Yolo-Fastest V2+ALA	352*352	0.11G	44.43(+1.53)
Yolo-Fastest V2 +LE-FPN+GKC	352*352	0.11G	47.26(+4.36)
***Yolo-Fastest V2**	352*352	0.11G	49.92(+7.02)

GKC: Grouping K-means Clustering; LE-FPN: Location-enhanced FPN; ALA: Adaptive Label Assignment; *Yolo-Fastest V2: the improved Yolo-Fastest V2
The bold entity is different from other models in order to highlight the improved model.

TABLE IV
VALIDATION OF THE PROPOSED METHOD ON YOLO-FASTEST V2 (UA-DETRAC)

Method	Size	All	car	bus	van	others
Yolo-Fastest V2	352*352	40.00	60.20	67.50	26.20	6.15
***Yolo-Fastest V2**	352*352	41.97(+1.97)	60.60(+0.40)	67.60(+0.10)	27.30(+1.10)	12.40(+6.25)

The bold entity is different from other models in order to highlight the improved model.

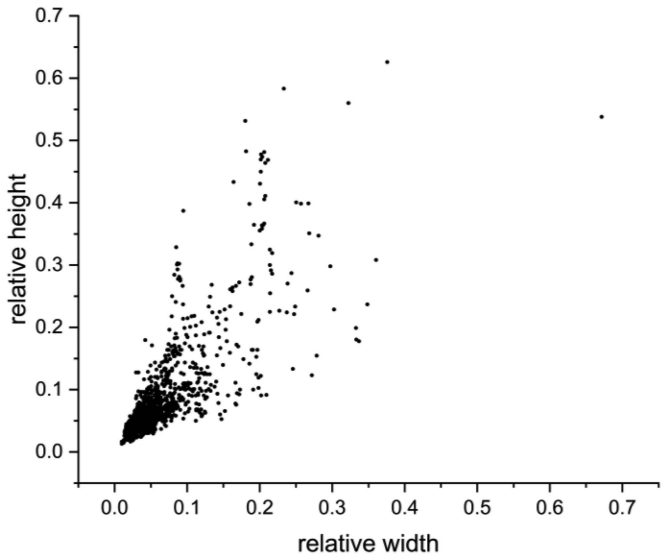

Fig. 9. Relative width and height distribution of the objects in the training set.

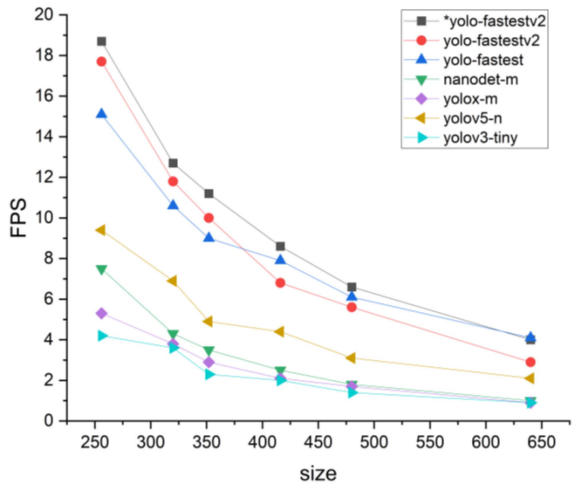

Fig. 10. Recognition speed comparisons of different sizes on validation dataset.

B. Ablation Experiment

1) Impact of Image Size: To achieve the goal of real-time detection, this article investigates the impact of different input image sizes on the recognition speed of the model. The picture size is divided into 256∗256, 320∗320, 352∗352, 416∗416, 480∗480, and 640∗640. For a fair comparison, all models are embedded on Raspberry Pi 4B 2018, and the recognition speed is calculated using NCNN. ∗Yolo-Fastest V2 is an improved model. As shown in Fig. 10, the recognition speed curve of ∗Yolo-FastestV2 is similar to the original model, demonstrating that the proposed method has no significant effect on the model's recognition. Video surveillance recognition mostly uses edge computing devices with limited computing power in real world. Because edge devices with limited computing power tend to use smaller image inputs and we hope the mAP50 to be as high as possible, the experimental standard for image input is set at 352∗352 on the Raspberry Pi.

2) Impact of the Proposed Method: Ablation experiments were conducted for Yolo-Fastest V2 to verify the effect of each proposed method on mAP50. Table III presents that introducing location-enhanced FPN and grouping k-means clustering strategies can increase mAP50 by 1.01% and 2.04%, respectively. The experimental results indicate that the feature fusion in the location-enhanced FPN improves the model's ability to

TABLE V
VALIDATION OF THE PROPOSED METHOD ON YOLOV5-N (PASCAL VOC DATASET)

Method	Size	FLOPs	mAP50
yolov5-n	352*352	0.63G	74.00
yolov5-n+LE-FPN	352*352	0.63G	74.10(+0.10)
yolov5-n+GKC	352*352	0.63G	74.50(+0.50)
yolov5-n+ALA	352*352	0.63G	74.90(+0.90)
yolov5-n+LE-FPN+GKC	352*352	0.63G	74.80(+0.80)
***yolov5-n**	352*352	0.63G	75.10(+1.10)

LE-FPN: Location-enhanced FPN. *yolov5-n: the improved yolov5-n.
The bold entity is different from other models in order to highlight the improved model.

TABLE VI
MODEL COMPARISON

Model	Size	FLOPs	Parameters	Speed (NCNN)	mAP50
YOLOV3-Tiny	352*352	1.97G	8.68M	324.08ms/3.89FPS	40.50
YOLOX-Nano	352*352	2.08G	2.24M	343.92ms/2.91FPS	40.37
YOLOV5-n	352*352	4.20G	1.27M	337.77ms/2.96FPS	67.30
NanoDet-m	352*352	0.41G	937.23K	303.40ms/3.30FPS	46.83
Yolo-Fastest	352*352	0.16G	291.42K	158.22ms/6.32FPS	38.50
Yolo-Fastestv2	352*352	0.11G	237.55K	100.41ms/9.96FPS	42.90
***Yolo-Fastestv2**	352*352	0.11G	342.95K	88.95ms/11.24FPS	49.92

The bold entity is different from other models in order to highlight the improved model.

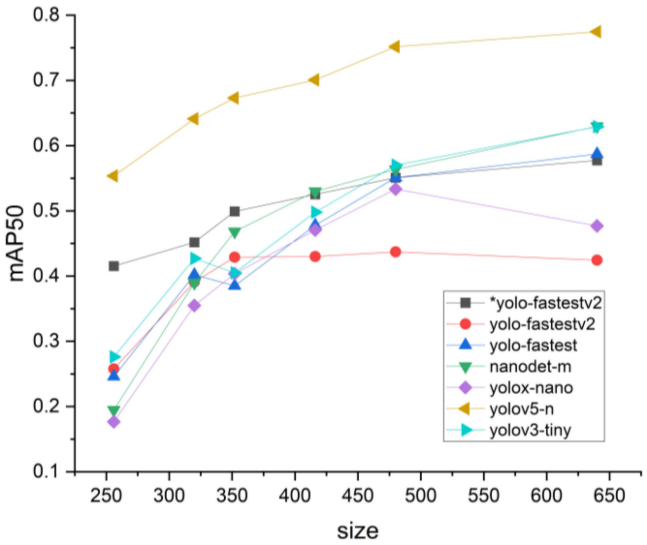

Fig. 11. mAP50 comparisons of different sizes on validation dataset.

learn features. The grouping k-means clustering strategy contributes positively to assigning an appropriate anchor box to each small-scale object. Replacing the original shape matching in Yolo-FastestV2 with the proposed adaptive label assignment leads to a 1.53% improvement in mAP50. The experimental results demonstrate that adaptive label assignment ensures that each object is assigned to an anchor with a similar shape and high overlap. It ensures the model is trained with a large number of positive examples. The two methods of location-enhanced FPN and grouping k-means clustering strategy improve the mAP50 by 4.36%. Together, the three proposed methods increase mAP50 by 7.02%, proving that the combined scheme is effective.

To demonstrate the robustness of the proposed method, it is implemented within the yolov5-n model and validated using the Pascal VOC (VOC2007+VOC2012) dataset. The experimental results are shown in Table V. The FPN structure of yolov5-n increases the mAP50 by 0.10% after adding the location enhancement method. The grouping k-means clustering strategy improves mAP50 by 0.50%. Adaptive label assignment increases the mAP50 by 0.90%. After applying the grouping k-means clustering, the location-enhanced structure improved mAP50 by 0.80%. Moreover, the method that combines location-enhanced and adaptive label assignment increases mAP50 by 1.10%. Consequently, the experimental results demonstrate the robustness of the proposed method.

C. Compared With State-of-the-Art Models

To evaluate the impact of the proposed method on the model's ability to extract features, six current state-of-the-art lightweight detectors are compared with the model that has been enhanced by the proposed method. Among the six most advanced methods,

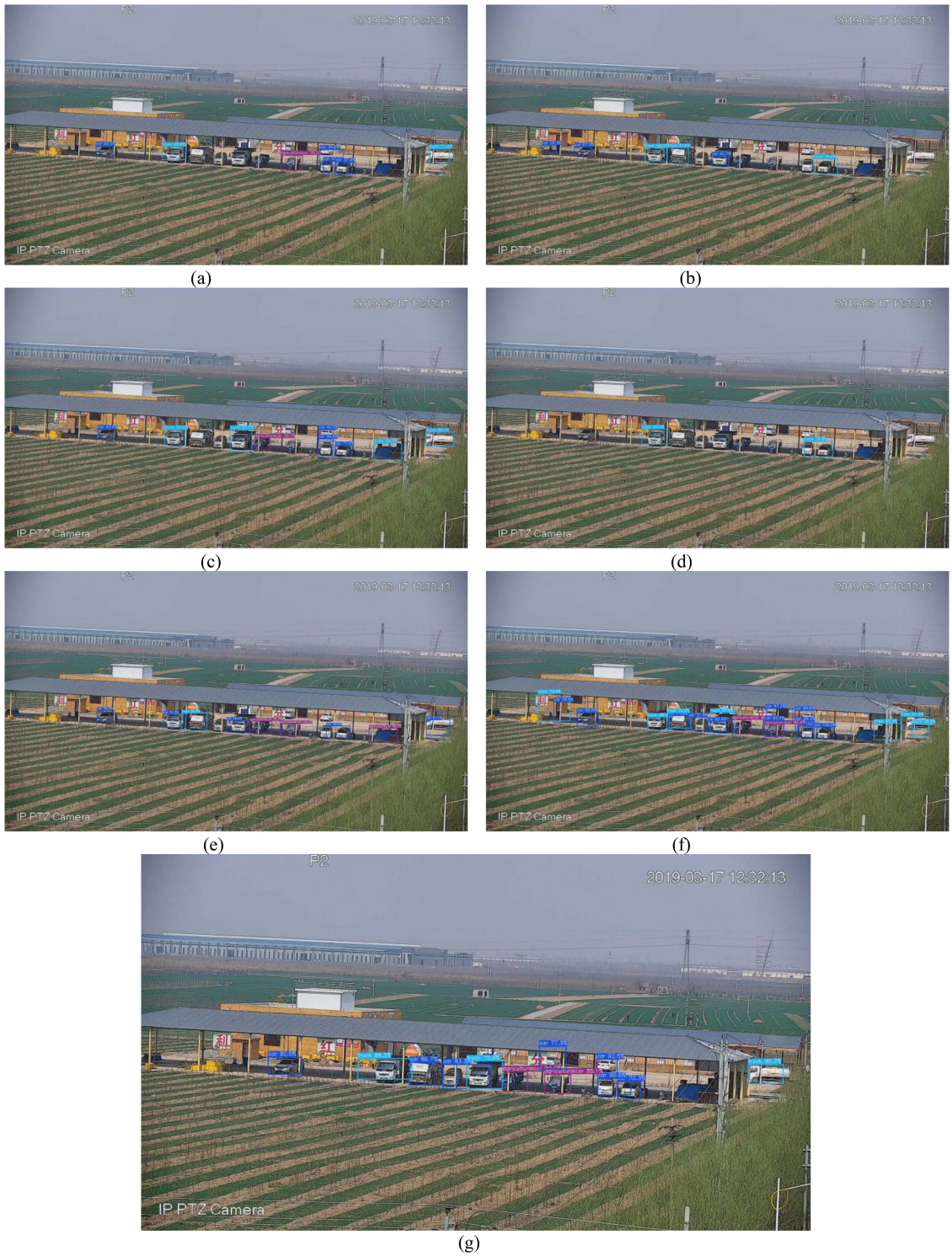

Fig. 12. Comparison of the NCNN inference computing results on Raspberry Pi 4B. (a) YOLOV3-tiny. (b) Yolo-Fastest. (c) YOLOV5-n. (d) YOLOX-Nano. (e) Nanodet-m. (f) Yolo-FatsestV2. (g) Improved Yolo-Fastest V2.

Yolo-Fastest and Yolo-FastestV2 are lightweight from YOLOV3 and YOLOV5, respectively. The others are official lightweight models.

Compared with the original model Yolo-FastestV2, the proposed method can enhance the model's ability to extract feature. Fig. 11 demonstrates that when the input image size exceeds 352, the mAP50 curve of Yolo-FastestV2 tends to be flat. At this point, the capability of feature extraction reaches a bottleneck. Compared with the improved model of the proposed method, the model's ability to extract features is enhanced. After the improved model, the mAP50 curve still maintains an increasing trend when the input image size is greater than 352. When the

Fig. 13. Detections for different scenes on Raspberry Pi 4B.

input image size is less than or equal to 352, the improved model has a higher mAP50 than the model with the same parameter level. When the input image size exceeds 352, the improved model exhibits a slightly lower mAP50 growth trend than NanoDet-m, YOLOV3-Tiny, and Yolo-Fastet. Due to the model size, it results in insufficient extraction of the object's feature information.

Table VI compares the models quantitatively in terms of FLOPs, parameter quantities, recognition speed, and mAP50. The proposed method increases the number of parameters by 105.4K but does not increase FLOPs. Specifically, compared with YOLOV3-Tiny, YOLOX-Nano, NanoDet-m, Yolo-Fastest, and Yolo-FastestV2, the proposed method increases mAP50 by 9.42%, 9.55%, 3.09%, 11.42%, and 7.02% for Yolo-FastestV2. It is 7.35 FPS, 8.33 FPS, 7.94 FPS, 4.92 FPS, and 1.28 FPS faster in terms of recognition speed. The improved model is 17.38% lower in mAP50 than YOLOV5-n, but 8.28 FPS faster. Therefore, in a tradeoff between speed and accuracy, the performance of the enhanced model is optimal. In Table IV, the experiment uses UA-DETRAC vehicle benchmark dataset, including car, bus, van, and others. The number of cars, bus, van, and others in the training set is 503 853, 57 051, 33 651, and 3726, respectively. The number of cars, bus, van, and others in the test set is 548 555, 38 519, 71 785, and 16 915, respectively. The proposed method increases car, bus, van, and others by 0.40% mAP50, 0.10% mAP50, 1.10% mAP50, and 6.25% mAP50, respectively, up 1.97%mAP50 overall. The experimental results show that this method can reasonably select positive training samples and fully learn the vehicle feature under video surveillance equipment.

To visually demonstrate the effect of vehicle detection in Raspberry Pi, the experimental comparison results for each model are presented in Fig. 12. The comparison of Fig. 12(f) and (g) shows that the proposed method identifies fewer errors and makes bbox positioning more accurate. This demonstrates that our method fully learns the location features and semantic features of small objects under video surveillance. In Fig. 12(a)–(f), other detectors have the phenomenon of missing detection. However, Fig. 12(g) shows that the number of missing detections is less for the improved Yolo-Fastest V2. This demonstrates our method can choose better positive training samples in order to transmit an effective gradient to the model during the forward propagation. This can fully learn the effective features of the object. The recognition results of other scenes are shown in Fig. 13 for the improved Yolo-Fastest V2.

IV. Conclusion

To overcome the phenomenon of low recognition rate and inaccurate localization of bbox when identifying small-scale engineering vehicles in surveillance videos using lightweight object detection models in embedded devices, this article proposes a method that combines location enhancement and adaptive label assignment. This method prevents a decrease in model recognition speed and improves model recognition performance. It is crucial for the monitor to identify engineering vehicles that illegally transfer land, as this can reduce government departments' human and financial resources. The methods include location-enhanced FPN structure, grouping k-means clustering strategy, and adaptive label assignment. The location-enhanced FPN structure utilizes concatenate and addition feature fusion to fuse the shallow and deep layer feature maps together. This structure enables the deep feature maps to contain rich location information and improves the model's capacity to locate the bboxes of small objects. Second, the grouping k-means clustering strategy is capable of obtaining multiscale anchors for small objects. Finally, the proposed adaptive label assignment

algorithm selects an appropriate anchor for each object to be identified and assigns the anchor to a label.

Tested in the monitoring dataset of engineering vehicles, this method improves the recognition accuracy of the original model from 42.90% mAP50 to 49.92% mAP50. The number of model parameters is increased by 105.5K, but there is no significant increase in FLOPs. The recognition speed in the Raspberry Pi can still reach 11.24FPS. To demonstrate that the proposed method has a certain degree of robustness, it is applied to yolov5-n. It provides a 1.10% mAP50 increase. It also demonstrates that the proposed method is applicable to both lightweight and large-scale models.

The proposed method still has some shortcomings. For instance, if the application scenario is modified, the model must perform k-means clustering again. Therefore, follow-up research will be improved in the direction of anchor sfree.

ACKNOWLEDGMENT

The authors would like to thank Pascal VOC (http://host.robots.ox.ac.uk/pascal/VOC/) and UA-DETRAC (https://detrac-db.rit.albany.edu) for making the object recognition datasets publicly available and the anonymous reviewers for their comments to improve this article.

REFERENCES

[1] X. Cui, S. Li, X. Wang, and X. Xue, "Driving factors of urban land growth in Guangzhou and its implications for sustainable development," *Front. Earth Sci.*, vol. 13, no. 3, pp. 464–477, Apr. 2019.

[2] D. Zhang, X. Liu, X. Wu, Y. Yao, X. Wu, and Y. Chen, "Multiple intra-urban land use simulations and driving factors analysis: A case study in Huicheng, China," *GISci. Remote Sens.*, vol. 56, no. 2, pp. 282–308, 2019.

[3] H. Wu, A. Lin, X. Xing, D. Song, and Y. Li, "Identifying core driving factors of urban land use change from global land cover products and POI data using the random forest method," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 103, 2021, Art. no. 102475.

[4] X. Zhang, J. Song, Y. Wang, W. Deng, and Y. Liu, "Effects of land use on slope runoff and soil loss in the Loess Plateau of China: A meta-analysis," *Sci. Total Environ.*, vol. 755, Feb. 2021, Art. no. 142418.

[5] H. Zhang, Z. Wang, B. Yang, J. Chai, and C. Wei, "Spatial-temporal characteristics of illegal land use and its driving factors in China from 2004 to 2017," *Int. J. Environ. Res. Public Health*, vol. 18, no. 3, Feb. 2021, Art. no. 1336.

[6] X. Xiang, N. Lv, X. Guo, S. Wang, and A. El Saddik, "Engineering vehicles detection based on modified faster R-CNN for power grid surveillance," *Sensors*, vol. 18, no. 7, Jul. 2018, Art. no. 2258.

[7] Y. Yang et al., "A fast and effective video vehicle detection method leveraging feature fusion and proposal temporal link," *J. Real-Time Image Process.*, vol. 18, no. 4, pp. 1261–1274, May 2021.

[8] S. Sri Jamiya and P. Esther Rani, "An efficient method for moving vehicle detection in real-time video surveillance," in *Advances in Smart System Technologies*. Singapore: Springer, 2021, pp. 577–585.

[9] A. Fedorov, K. Nikolskaia, S. Ivanov, V. Shepelev, and A. Minbaleev, "Traffic flow estimation with data from a video surveillance camera," *J. Big Data*, vol. 6, no. 1, Aug. 2019, Art. no. 73.

[10] A. B. Mabrouk and E. Zagrouba, "Abnormal behavior recognition for intelligent video surveillance systems: A review," *Expert Syst. Appl.*, vol. 91, pp. 480–491, Jan. 2018.

[11] J. Lim, M. I. Al Jobayer, V. M. Baskaran, J. M. Y. Lim, J. See, and K. Wong, "Deep multi-level feature pyramids: Application for non-canonical firearm detection in video surveillance," *Eng. Appl. Artif. Intell.*, vol. 97, Jan. 2021, Art. no. 104094.

[12] F. Pérez-Hernández, S. Tabik, A. Lamas, R. Olmos, H. Fujita, and F. Herrera, "Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance," *Knowl.-Based Syst.*, vol. 194, Apr. 2020, Art. no. 105590.

[13] J. Sun, J. Shao, and C. He, "Abnormal event detection for video surveillance using deep one-class learning," *Multimedia Tools Appl.*, vol. 78, no. 3, pp. 3633–3647, Sep. 2019.

[14] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu, and R. S. M. Goh, "Anomalynet: An anomaly detection network for video surveillance," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 10, pp. 2537–2550, Oct. 2019.

[15] J. T. Zhou, L. Zhang, Z. Fang, J. Du, X. Peng, and Y. Xiao, "Attention-driven loss for anomaly detection in video surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 12, pp. 4639–4647, Dec. 2020.

[16] R. Nawaratne, D. Alahakoon, D. De Silva, and X. Yu, "Spatiotemporal anomaly detection using deep learning for real-time video surveillance," *IEEE Trans. Ind. Inform.*, vol. 16, no. 1, pp. 393–402, Jan. 2020.

[17] K. Muhammad, J. Ahmad, Z. Lv, P. Bellavista, P. Yang, and S. W. Baik, "Efficient deep CNN-based fire detection and localization in video surveillance applications," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 7, pp. 1419–1434, Jul. 2019.

[18] H. Wei, M. Laszewski, and N. Kehtarnavaz, "Deep learning-based person detection and classification for far field video surveillance," in *Proc. IEEE 13th Dallas Circuits Syst. Conf.*, 2018, pp. 1–4.

[19] I. M. Nasir, M. Raza, J. H. Shah, S.-H. Wang, U. Tariq, and M. A. Khan, "HAREDNet: A deep learning based architecture for autonomous video surveillance by recognizing human actions," *Comput. Elect. Eng.*, vol. 99, Apr. 2022, Art. no. 107805.

[20] M. A. Khan et al., "Human action recognition using fusion of multiview and deep features: An application to video surveillance," *Multimedia Tools Appl.*, pp. 1–27, Mar. 2020, doi: 10.1007/s11042-020-08806-9.

[21] G. Sreenu and M. A. S. Durai, "Intelligent video surveillance: A review through deep learning techniques for crowd analysis," *J. Big Data*, vol. 6, no. 1, Jun. 2019, Art. no. 48.

[22] M. Shorfuzzaman, M. S. Hossain, and M. F. Alhamid, "Towards the sustainable development of smart cities through mass video surveillance: A response to the COVID-19 pandemic," *Sustain. Cities Soc.*, vol. 64, Jan. 2021, Art. no. 102582.

[23] H.-H. Tseng, M.-D. Yang, R. Saminathan, Y.-C. Hsu, C.-Y. Yang, and D.-H. Wu, "Rice seedling detection in UAV images using transfer learning and machine learning," *Remote Sens.*, vol. 14, no. 12, Jun. 2022, Art. no. 2837.

[24] J. Lee, J. Wang, D. Crandall, S. Šabanović, and G. Fox, "Real-time, cloud-based object detection for unmanned aerial vehicles," in *Proc. 1st IEEE Int. Conf. Robot. Comput.*, 2017, pp. 36–43.

[25] A. Anjum, T. Abdullah, M. F. Tariq, Y. Baltaci, and N. Antonopoulos, "Video stream analysis in clouds: An object detection and classification framework for high performance video analytics," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1152–1167, Oct./Dec. 2019.

[26] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[27] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "A survey on mobile augmented reality with 5G mobile edge computing: Architectures, applications, and technical aspects," *IEEE Commun. Surv. Tut.*, vol. 23, no. 2, pp. 1160–1192, Apr./Jun. 2021.

[28] N. Balamuralidhar, S. Tilon, and F. Nex, "MultEYE: Monitoring system for real-time vehicle detection, tracking and speed estimation from UAV imagery on edge-computing platforms," *Remote Sens.*, vol. 13, no. 4, Feb. 2021, Art. no. 573.

[29] P. Gupta, B. Pareek, G. Singal, and D. V. Rao, "Edge device based military vehicle detection and classification from UAV," *Multimedia Tools Appl.*, vol. 81, no. 14, pp. 19813–19834, Jul. 2022.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[31] ultralytics/yolov5: YOLOv5 in PyTorch >ONNX >CoreML >TFLite (github.com), 2020.

[32] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," 2021, *arXiv:2107.08430*, doi: 10.48550/arXiv.2107.08430.

[33] Y. Zhao, Y. Yin, and G. Gui, "Lightweight deep learning based intelligent edge surveillance techniques," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1146–1154, Dec. 2020.

[34] Z. Long, W. Suyuan, C. Zhongma, F. Jiaqi, Y. Xiaoting, and D. Wei, "Lira-YOLO: A lightweight model for ship detection in radar images," *J. Syst. Eng. Electron.*, vol. 31, no. 5, pp. 950–956, Oct. 2020.

[35] A. Ullah, K. Muhammad, W. Ding, V. Palade, I. Ul Haq, and S. W. Baik, "Efficient activity recognition using lightweight CNN and DS-GRU network for surveillance applications," *Appl. Soft Comput.*, vol. 103, May 2021, Art. no. 107102.

[36] S. Y. Nikouei, Y. Chen, S. Song, R. Xu, B.-Y. Choi, and T. Faughnan, "Smart surveillance as an edge network service: From harr-cascade, SVM to a lightweight CNN," in *Proc. IEEE 4th Int. Conf. Collab. Internet Comput.*, 2018, pp. 256–265.

[37] L. Steinmann, L. Sommer, A. Schumann, and J. Beyerer, "Fast and lightweight person detector for unmanned aerial vehicles," in *Proc. Eur. Signal Process. Conf.*, 2019, pp. 1–5.

[38] H. Zhao et al., "Mixed YOLOv3-LITE: A lightweight real-time object detection method," *Sensors*, vol. 20, no. 7, Mar. 2020, Art. no. 1861.
[39] RangiLyu/nanodet: NanoDet-Plus Super fast and lightweight anchor-free object detection model. Only 980 KB(int8) /1.8MB (fp16) and run 97FPS on cellphone (github.com), 2021.
[40] A. N. Amudhan and A. P. Sudheer, "Lightweight and computationally faster hypermetropic convolutional neural network for small size object detection," *Image Vis. Comput.*, vol. 119, Mar. 2022, Art. no. 104396.
[41] J. Liu, C. Hu, J. Zhou, and W. Ding, "Object detection algorithm based on lightweight YOLOv4 for UAV," in *Proc. 7th Int. Conf. Intell. Comput. Signal Process.*, 2022, pp. 425–429.
[42] dog-qiuqiu/Yolo-FastestV2: Based on Yolo's low-power, ultra-lightweight universal target detection algorithm, the parameter is only 250k, and the speed of the smart phone mobile terminal can reach ∼300fps+ (github.com), 2022.
[43] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9756–9765.

Hairong Zhang received the Ph.D. degree in geodesy and survey engineering from the China University of Mining and Technology, Xuzhou, China, in 2002.

He is currently an Associate Professor with the School of Environment and Spatial Informatics, China University of Mining and Technology, Xuzhou, China. He is also a Guest Professor with the Key Laboratory of Natural Resources Monitoring in Tropical and Subtropical Area of South China, Ministry of Natural Resources, Guangzhou, China. His research interests include GIS, deep learning, and geographical modeling.

Dongsheng Xu received the B.S. degree in human geography and urban and rural planning from the School of Surveying and Land Information Engineering, Henan Polytechnic University, Jiaozuo, China, in 2020. He is currently working toward the M.S. degree in cartography and geographical information engineering with the China University of Mining and Technology, Xuzhou, China.

His research interests include deep learning and model compression.

Dayu Cheng received the Ph.D. degree in cartography and geographic information engineering from the China University of Mining and Technology, Xuzhou, China, in 2012.

He is currently an Associate Professor with the School of Mining and Geomatics Engineering, Hebei University of Engineering, Handan, China. His research interests include geographic big data mining, computing vision, and GIS development and applications.

Xiaoliang Meng received the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2009.

He is currently a "Luo Jia Distinguished Professor" with Wuhan University. He is also a Guest Professor with the Key Laboratory of Natural Resources Monitoring in Tropical and Subtropical Area of South China, Ministry of Natural Resources, Guangzhou, China. His research interests include intelligent sensor network and geospatial information service in the field of ecology and environment.

Dr. Meng was the recipient of the "Best Young Authors Award" of the International Society for Photogrammetry and Remote Sensing in 2012.

Geng Xu received the B.S. degree in geophysics in 2020 from the School of Resources and Geosciences, China University of Mining and Technology, Xuzhou, China, where he is currently working toward the M.S. degree in cartography and geographical information engineering with the China University of Mining and Technology, Xuzhou, China.

His research interests include mine water inrush intelligent recognition with multifield coupling based on deep learning.

Wei Liu received the M.S. and Ph.D. degrees in cartography and geographic information engineering from the China University of Mining and Technology, Xuzhou, China, in 2007 and 2010, respectively.

He is currently an Associate Professor with the School of Geography, Geomatics and Planning, Jiangsu Normal University, Xuzhou, China. His research interests include spatial data quality checking, high-resolution remote sensing image processing, and GIS development and applications.

Teng Wang received the master's degree in telecommunications engineering and telecommunication networks from the University of Technology Sydney, Ultimo, NSW, Australia, in 2014.

He is currently a Senior Engineer with the Lands and Resource Department of Guangdong Province, Surveying and Mapping Institute, Guangzhou, China. His research interests include GIS and computing vision.