Object Detection for UAV Images Based on Improved YOLOv6

Yingwei Li, Xiaoxia Zhang

Abstract—**This paper, we have improved the YOLOv6 object detection network to better detect high-density, small target objects in UAV aerial images. First of all, we designed a mixed data augmentation method which adds background samples to the training, so that the model can better identify falsely detected and falsely detected targets without changing the computational complexity. In addition, to prevent the loss of feature information caused by small target features in the upsampling stage, we have strengthened the Feature Pyramid Networks (FPN) with Feature Alignment Module (FAM) and Feature Selection Module (FSM). Finally, we have formed the Transformer Prediction Head (TPH) by using the Transformer Encoder Block to replace the prediction head of the original model. This ensures accurately locate targets in high-density scenes. The experiment used the Visdrone2019-DET drone aerial photography dataset. Compared with the original YOLOv6 network, the average precision (AP50) of the improved network on the verification set when the IoU is 0.5 was 59.73%, which is 6.02% higher than before the improvement. Comparison results with other mainstream object detection algorithms show that the performance of our proposed algorithm is better than other detectors. Therefore, the algorithm proposed in this paper can effectively detect objects with high density and small targets in UAV aerial photography scenes.**

Index Terms—**Object detection, YoloV6, Transformer Encoder Block, Visdrone2019-DET, IoU.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) have a long history, dating back to World War I. In an effort to reduce casualties, the British developed small aircraft that were controlled by radio rather than human pilots, allowing them to fly to a certain target to complete specific tasks. UAVs possesses unique characteristics such as flexible maneuverability, high efficiency and speed, precision and accuracy, low operating costs, wide application range, and short production cycles. As a result, they are widely used in various fields including agricultural plant protection, disaster rescue, field monitoring, and power inspection [1].

Computer vision refers to use of computers to replace the human eyes. Object detection, based on deep learning, is one of the core problems in the field of computer vision. It involves identifying all objects of interest in an image, determining their categories and positions. A large number of excellent object detection methods have been developed as yet. The YOLO series algorithm [2]-[5], representing the One-stage detector, has attracted extensive attention of researchers due to its powerful detection performance and faster detection speed. The One-Stage detector initially used the Anchor-based mechanism to classify the target and determine the bounding box coordinate regression. However, scholars later proposed the idea of Anchor-free detection, which minimizes the computing power required by the model without loss of detection performance.

Aerial image object detection is a vital technology combining UAV and deep learning, which plays an important role in civil and national defense. Nazia Attari et al. [6] artificially assessed the degree of damage after the PAM hurricane, and proposed the Nazr-CNN deep learning network model, which applied the deep learning of object detection and the fine-grained classification to images captured by drones. Nazr-CNN consists of two parts, the function of the first component is to locate objects (such as houses) in images by performing pixel-level classification. In the second component, Fisher Vectors (FV) of the fragments generated by the first component are encoded using a convolutional neural network (CNN) to help distinguish between different levels of damage. Seyed Majid Azimi [7] proposed ShuffleDet to detect vehicles in images captured by drones. Channel shuffling and grouped convolutions are used to reduce the computational cost of the model, and is evaluated on the CARPK and PUCPR+ datasets, running at 14 frames per second on an NVIDIA Jetson TX2. The CenterNet proposed by Kaiwen Duan et al. [8] regarded the positioning task as the task of detecting the center point and its offset, and employed a predicted focal point-based regression method to derive the actual position information from the offset parameters regressed from the center. This method effectively enhanced the detection rate of small targets in images captured by UAV, but the high-resolution output also increased the reasoning delay. The breakthrough of Transformer in the field of computer vision has made the combination of CNN and Transformer a hot topic [9]. Xingkui Zhu et al. [10] introduced Transformer to YOLOv5, improving the network's performance in detecting small target objects.

II. Related Work

A. YoloV6 Algorithm Principle

Object detection algorithms can be broadly classified into two categories: One-Stage and Two-Stage. One-Stage algorithms require only a single pass of the original image through the network to predict all bounding boxes, while

Manuscript received December 18, 2022; revised April 20, 2023.

Y. W. Li is a postgraduate student of School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan 114051, China (e-mail: 1294999811@qq.com).

X. X. Zhang is a Professor of School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan 114051, China (phone: 86-0412-5929812; e-mail: aszhangxx@163.com).

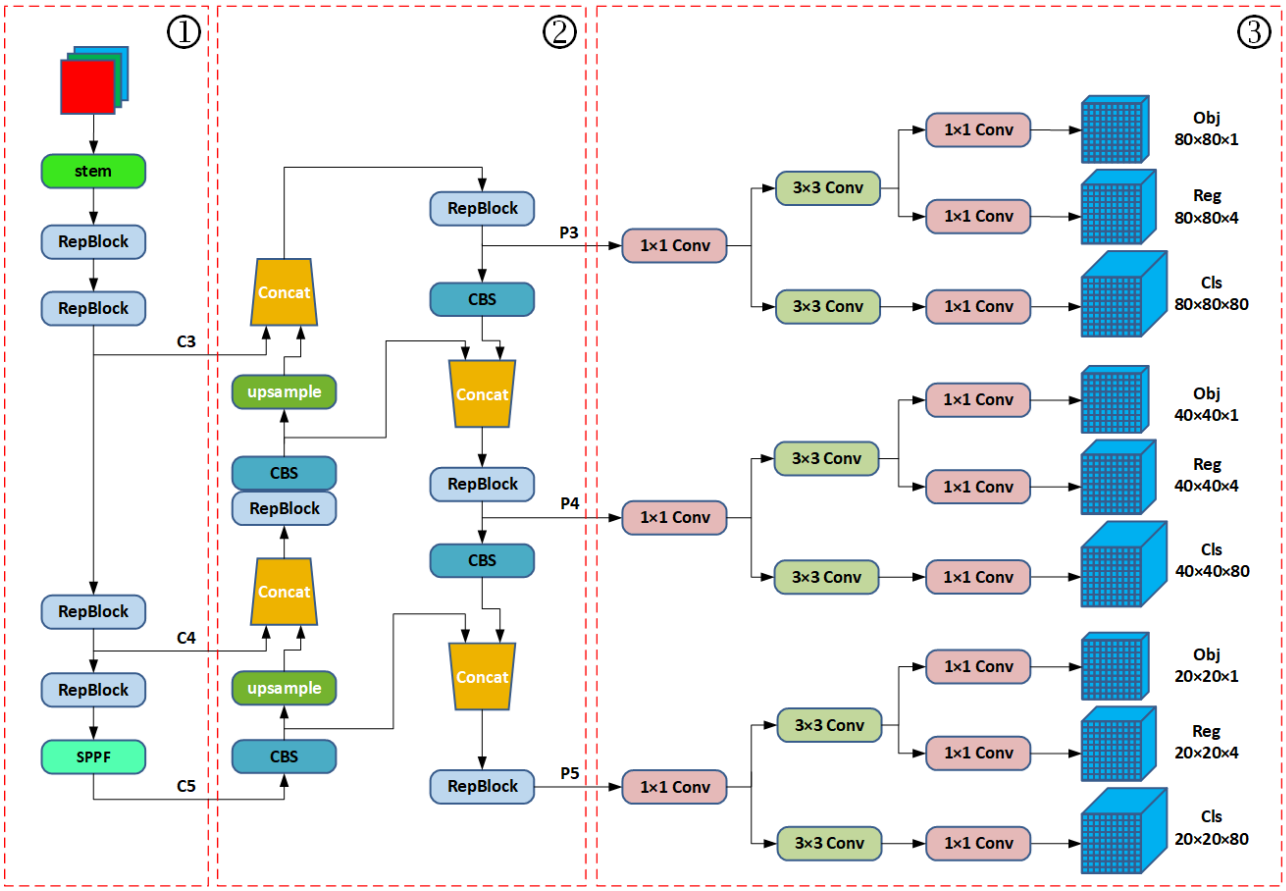


Fig. 1. YoloV6 network architecture.

Two-Stage algorithms generate candidate boxes for areas that may contain objects before classifying each candidate box. Therefore, One-Stage algorithms have an inherent advantage in detection speed, leading to a lot of research focusing on their improvement. The YOLO series algorithm is one of the popular One-Stage algorithms, widely used in industrial-level object detection applications. Following YOLOv5, the Meituan technical team has launched a new object detection algorithm YOLOv6 [11].

Compared to YOLOv5, YOLOv6 has improved in many aspects such as Backbone, Neck, Head, loss function, quantitative deployment, and training strategy. It provides five pre-training models for different industrial application scenarios, namely YOLOv6-N, YOLOv6-T, YOLOv6-S, YOLOv6-M, and YOLOv6-L. The overall network structure of YOLOv6 is shown in Fig. 1

The first part in Fig. 1 is the EfficientRep Backbone network structure proposed by YOLOv6, in which four series-connected RepBlocks are used inside the network for feature extraction. The last three layers of the network output three branches of feature extraction results, respectively C3, C4 and C5, which are input into the Neck. RepBlock is one of the most important structures in EfficientRep Backbone and is inspired by the repVGG [12] network. It uses the advantages of multi-branch structure during the training phase to make the overall feature extraction phase has higher performance. In the reasoning phase, the multi-branch structure is equivalently converted to a single-path structure, resulting in memory savings during inference and improved efficiency. The network structure improves the performance of feature extraction on the basis of accelerating the calculation speed and saving the operation cost.

During the training phase, the RepBlock structure is shown in Fig. 2. Each RepBlock comprises a three-branch structure, with an additional parallel 1×1 convolution branch and an identity mapping branch added into each convolutional layer, and finally performing element-wise addition. The 3×3 convolution is highly efficient on mainstream GPUs and CPUs, thereby making the detection network faster and stronger.

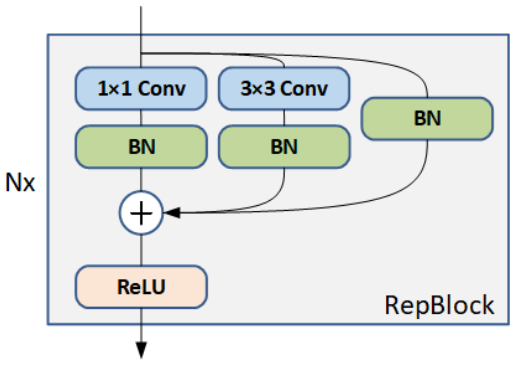


Fig. 2. The structural re-parameterization backbone during training.

In the inference stage, to maximize hardware computing power utilization, the 3×3 convolution in the multi-branch model is fused with the BN layer, and converted into a new 3×3 convolution single-path model with a ReLU activation function, which constitutes the RepConv module, is shown in Fig. 3.

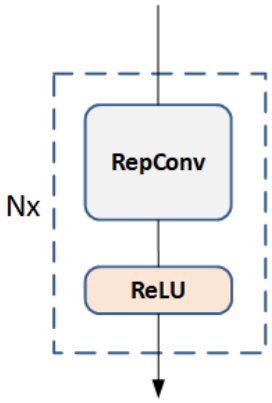


Fig. 3. The structural re-parameterization backbone during inference.

The second part in Fig. 1 is the Rep-PAN network structure proposed by YOLOv6, and the Neck overall presents a "U"-shaped structure, with the left side comprising an upsampling of the feature map, gradually expanding its width and height. The right side is a bottom-up path, and the output feature map from the left side is channel-concatenated to facilitate two-way feature fusion. The CBS module comprises a convolution block, a BN layer, and a SiLU activation function. A size of 1×1 convolution block is used on the left side of the structure, whilst a 3×3 convolution block is used on the right side.

The third part in Fig. 1 is the decoupled head structure proposed by YOLOv6, which is compared with the prediction head of other state-of-the-art object detection models, such as YOLOv5, FCOS [13], YOLOX [14], etc. Each prediction head in YOLOv5 is tasked with both classification and regression, which are realized through branch fusion sharing. However, as the regions of interest for these tasks differ, which is not conducive to the improvement of the effect. The decoupled head used in FCOS and YOLOX separates the classification and regression tasks, and used different branches to complete separate computing tasks. While multiple 3×3 convolutions are used inside the structure, this can lead to improved accuracy at the expense of increased network delay. YOLOv6 has streamlined the design of the prediction head, comprehensively considers the balance between the representational ability of related operators and the hardware computing overhead, and Hybrid Channels strategy is used to redesign a more efficient decoupled head structure.

Furthermore, to accommodate the usage scenarios in both academy and industry, YOLOv6 has adopted several latest technologies, including self-distillation, TAL and SimOTA label allocation technology, Anchor-free, SIoU regression loss, RepOptimizer-QAT, etc. Overall, YOLOv6 far exceeds other object detection algorithms in terms of detection accuracy and speed.

B. Vision Transformer Algorithm Principle

Transformer was initially developed for natural language processing. In 2020, the Google team proposed the Vision Transformer (ViT) [15] algorithm, which applies the field of image classification. This breakthrough innovation attracted widespread attention from scholars at home and abroad. Fig. 4 shows the Vision Transformer model structure diagram.

The input of Vision Transformer draws on its ideas in the field of natural language processing, splits the 224×224 input image into several small image patches, and expands all small image patches into 256-length vectors, forming a linear projection of flattened patches. After normalization by Multilayer Perceptron (MLP), a series of one-dimensional vectors will be obtained, which is the input. In order to prevent the split image patches from losing position information, position information is also embedded for each image patches, and finally the combined content is input to Transformer Encoder.

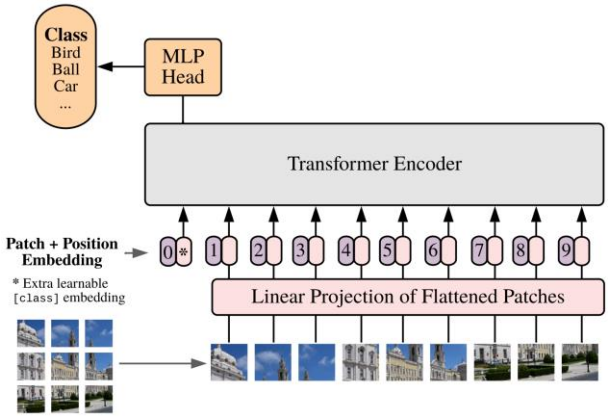


Fig. 4. Vision Transformer model structure diagram.

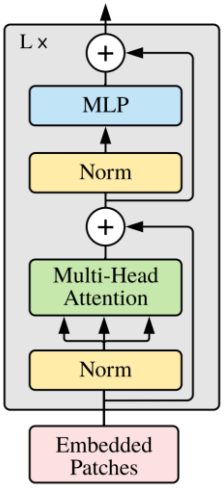


Fig. 5. Transformer Encoder model structure diagram.

The model structure of Transformer Encoder is shown in Fig. 5, which mainly consists of two sub-layer structures. The first sublayer is the multi-head attention mechanism sublayer, which consists of multiple attention mechanisms. Each attention mechanism structure generates three vectors Q, K, and V, where Q is a single vector, K and V are two sets of vectors. Its function is to map the Q vector into a new vector, obtained by weighting all the vectors in the V set. The weighting coefficient of each vector in the V set is calculated by the corresponding vector in the Q vector and the K set. The new vectors output by each attention mechanism are concatenated together to form the final multi-head attention mechanism structure. This structure can combine different features for learning, similar to the role of multiple convolution kernels in CNN. The second sublayer is composed of a MLP, which is mainly used to classify

nonlinear problems. In a study by Yihe Dong et al. [16], they pointed out that without skip connect and MLP architecture, the expressive ability of the self-attention mechanism network decays with the increase of depth, and MLP plays a very important role in the process of preventing the degradation from happening.

End Transformer Encoder, MLP Head is used to classify the output results. It is mainly composed of LaryerNorm and two fully connected layers, and it is linearly activated by the GELU activation function.

Transformer can convert the object detection task into a set prediction task, without requiring a large number of pre-selected boxes or cumbersome NMS, and obtains the prediction result directly.

III. Improvement Strategy

A. *Background sample data augmentation method*

When using traditional deep learning object detection algorithm to detect aerial images, the situation of missed and false detection is extremely high. This is due to the wide shooting range and long distance of the drone camera, which makes it difficult for the model to detect small and dense targets. Dense small targets face many challenges such as low resolution, limited extractable features, scarce samples, and uneven distribution. The importance of data augmentation for small target objects is becoming increasingly significant.

In YOLOv6, Mosaic [5] and Mixup [17] are utilized to perform data augmentation operations on images respectively. Mosaic draws on the practice of Cutmix, randomly selecting 4 pictures from the training set each time, and then cutting and stitching them at random positions to form a new picture, which greatly enriches the detection background. Mixup is a mixed-class data augmentation algorithm, that combines the mixing coefficients calculated from the beta distribution of different classes of images to expand the dataset.

During the training process using the original dataset, there are often many false detections, and the target result of the false detection is not what we need. To address this issue, we draw on the advantages of Mosaic and Mixup, and propose a mixed data augmentation of background samples method, called MDAB. As shown in Fig. 6, in the detection results on the left, the circle position is detected as the "van" target and the "car" target, but by zooming in on the area, we can clearly distinguish, the "van" target is actually the road, and the "car" target is actually the roof of the bus. Therefore, pictures that do not contain detection targets, such as those on the right side of Fig. 6, are suitable for training as background samples, allowing the network model to further learn which targets need to be detected and which targets need to be excluded. Recognizing the detection target can help the network model reduce the rate of false detection and improve the robustness of small target detection tasks.

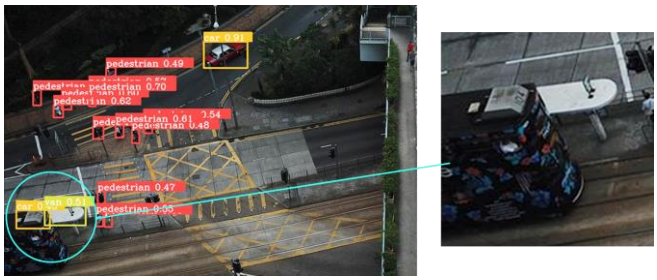
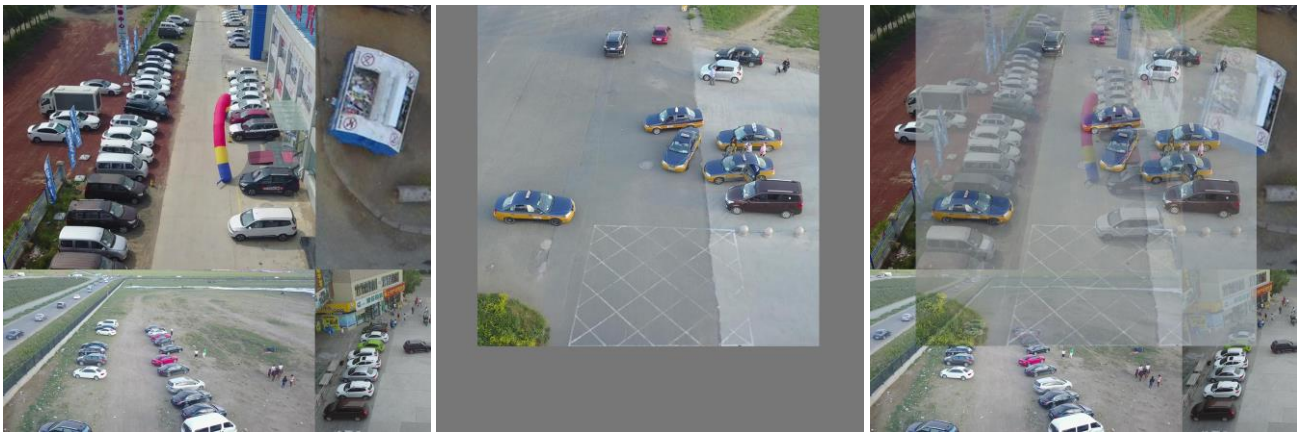


Fig. 6. Model false detection visualization results.

In summary, the main data source of the background sample set for MDAB is the area of the false detection in the cropped training image. This cropped area only contains the false detection targets and does not include the targets to be detected. The specific workflow of MDAB is as follows: first, 4 pictures with targets are randomly selected from the training set, and 1 picture is randomly selected from the background sample set. Mosaic data augmentation is applied to the 3 pictures with targets and the background sample image, resulting in a new image as shown in Fig. 7(a). Then, Mixup data augmentation is performed on this new image and the remaining one with the target picture in Fig. 7(b), to obtain the mixed data augmentation of background sample shown in Fig. 7(c).

B. *Strengthen the Feature Pyramid Networks*

For the image detection of dense small objects, two solutions are currently available. The first is to use dilated convolutions, which expand the convolution filters to capture longer-distance information. The second is to use a pyramid structure to better integrate features of different scales and obtain more semantic information.



(a) Mosaic data augmentation sample (b) A training sample (c) Mixed data sample

Fig. 7. Mixed data augmentation of background sample.

YOLOv6 uses the Rep-PAN pyramid structure to ensure that the model's prediction of features at different scales. However, it also continuously aggravates the loss of boundary details due to repeated downsampling operations. When upsampling, it can cause misalignment of contextual feature maps. Thus, directly using the fusion method of element-wise addition and channel-wise concatenation in the pyramid structure will further damage the prediction of boundary information, which ultimately results in misclassification of objects adjacent to the boundary.

We borrowed the idea of Feature Alignment Pyramid Network (FaPN) [18] and integrated the feature alignment into the Rep-PAN pyramid structure of the YOLOv6 network. FaPN structure is shown in Fig. 8. Compared with the Feature Pyramid Network (FPN) structure, FaPN adds a Feature Alignment Module (FAM) to the upsampling process and a Feature Selection Module (FSM) to the skip connection. FAM aligns an upsampled feature map with learned parameters to a set of reference feature maps by learning each sampling location in the convolution kernel and a learned offset. FSM uses the principle of attention mechanism to explicitly establish the dependency between network evolution and feature channels to improve the quality of network prediction

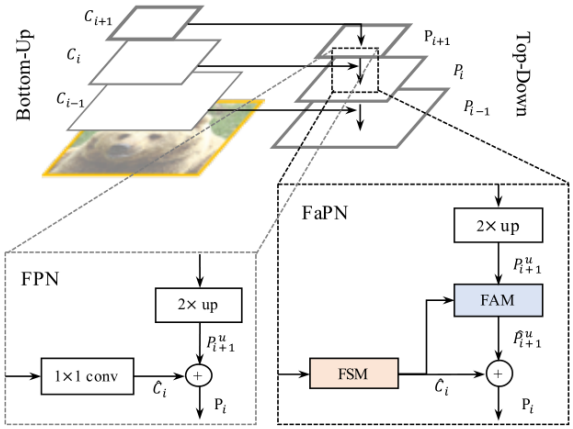


Fig. 8. FaPN structure diagram.

FAM consists mainly of two steps: the first step is to learn the spatial difference offset Δ_i according to the upsampled and downsampled feature maps, as shown in (1):

$$\Delta_i = f([\hat{C}_{i-1}, P_i^u]) \tag{1}$$

P_i^u represents the offset value after upsampling at each spatial position. $\hat{C}_{i-1}$ represents the correct spatial position information in Bottom-Up. Concatenate the $\hat{C}_{i-1}$ and P_i^u channels to obtain $[\hat{C}_{i-1}, P_i^u]$, which is used to represent the difference between the upsampled spatial position and the correct spatial position.

The second step is to act on the upsampling feature map $\hat{P}_i^u$ according to the deviation amount, as shown in (2):

$$\hat{P}_i^u = f(P_i^u, \Delta_i) \tag{2}$$

In (1) and (2), $f(\bullet)$ represents the function of correct location feature alignment with the learned offset. It is mainly composed of deformable convolution, which adds an offset to each point on the receptive field, and then dynamically adjusts the position of the upsampling alignment. Defined as follows:

$$f = \delta(F_1(z^h, z^w)) \tag{3}$$

To improve feature selection efficiently, FSM abandons the practice of using only 1×1 convolutional compression channel, and adopts a new form of squeeze and excitation to achieve precise distribution of spatial dimension feature maps.

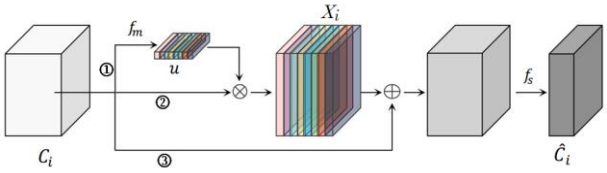


Fig. 9. FSM structure diagram.

FSM structure diagram is shown in Fig. 9. C_i represents the input three-dimensional feature map, and three branches are used to perform different operations on it. The branch 1 pair will squeeze the input features first, as shown in (4):

$$z_i = \frac{1}{H_i \times W_i} \sum_{h=1}^{H_i} \sum_{w=1}^{W_i} c_i(h, w) \tag{4}$$

c_i represents the two-dimensional matrix of the i-th channel in the input feature. H and W are the width and height of the feature map, respectively. Through (4), a squeezed output result of $1\times1\times i$ will be obtained. Next, the excitation is performed, and the weight of each feature after compression is recalculated using $f_m(\bullet)$, as shown in (5):

$$u = f_m(z_i) \tag{5}$$

In $f_m(\bullet)$, each squeezed feature weight is remodeled using 1×1 convolution and Sigmoid activation function, and the importance vector u is output. Next, use channel-wise multiplication, multiply the importance vector u with the original features to form branch 2, as shown in (6):

$$X_i = u \otimes C_i \tag{6}$$

The output result X_i is very important. It is used to describe the importance of each feature channel automatically obtained by fusing the feature channels and recalibrating the weight of the original feature map. Branch 3 makes element-wise addition of the original input features with X_i using skip connections, which is to avoid any specific channel response being over-amplified or suppressed.

Finally, to improve efficiency, the output result is further rescaled, as shown in (7).

$$\hat{C}_i = f_s(C_i + X_i) \tag{7}$$

A 1×1 convolution is used in $f_s(\bullet)$ to selectively keep important features and discard useless features for channel reduction. As a result, FSM is better applied to the Neck stage of the network, effectively enhancing multi-scale feature aggregation.

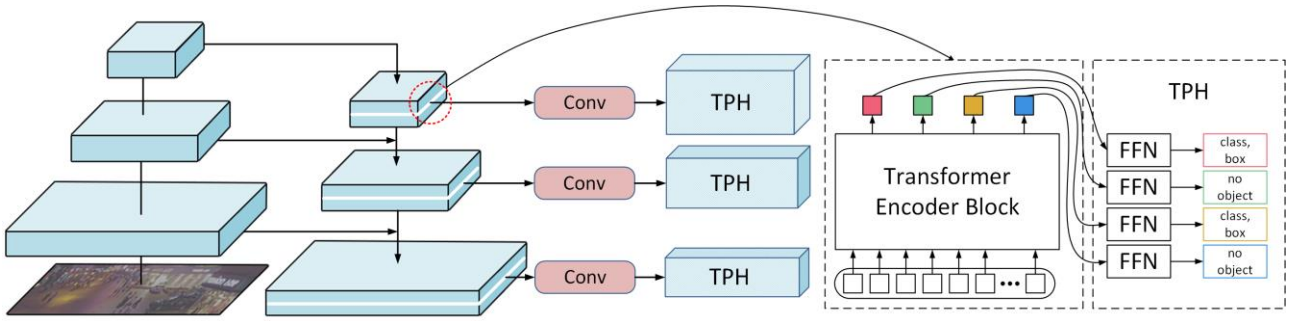


Fig. 10. TPH-YOLOv6 structure diagram.

C. Adjustment of Transformer Prediction Head

The traditional CNN object detection method relies on regression and classification to predict the target frame, which includes eliminating a large number of pre-selected boxes, anchors design, etc. These post-processing tasks will affect the prediction effect of the model. The application of Transformer in the field of computer vision also faces many challenges. The size of digital images is not as fixed as the token size in NLP tasks. It is more difficult to use Transformer to detect targets with variable sizes. In addition, due to the high image resolution, the computational complexity of using Transformer increases exponentially with the input feature size.

Regarding the shortcomings of the above two methods, we introduce the Transformer structure into the CNN, and use the CNN's feature map hierarchy to overcome the difficulty of the Transformer structure in predicting targets of varying sizes in images. The specific details of implementation are shown in Fig. 10. For the Rep-PAN network structure of YoloV6, each side can be divided into 3 layers, and we replace all RepBlock structures in the bottom-up path with the Transformer Encoder Block shown in Fig. 5. This is because the Transformer Encoder Block can capture global information and rich context information, and under the hierarchical feature map of the Rep-PAN network structure, it overcomes the difficulty of the Transformer structure in predicting targets of different sizes in the image.

In addition, we use the Transformer Encoder Block to construct the Transformer Prediction Head (TPH). Each TPH accepts the sequence information output from the Transformer Encoder Block, and its interior is composed of a shared Feed Forward Networks (FFN). Different from ordinary fully connected networks, FFN calculates the elements of each input sequence separately, and directly outputs the prediction results, avoiding the computational waste that arise from post-processing tasks in the CNN network.

IV. Experiment

A. Dataset Selection

Our experiments were carried out under the Visdrone2019-DET dataset [19], which was collected and marked by the AISKYEYE team of the Machine Learning and Data Mining Laboratory at Tianjin University. The dataset is captured by different types of drones in different scenes, different weather, and different lighting conditions, including outdoor scenes from 14 urban environments and rural environments separated by thousands of kilometers in China, sparse scenes and crowded scenes, etc. Fig. 11 shows the ratio distribution of the length and width of the annotated objects in the Visdrone2019-DET dataset with respect to the original image. It can be seen that the ratio of the length and width of most annotation objects to the original image is less than 0.1, so this dataset has rich small goals.

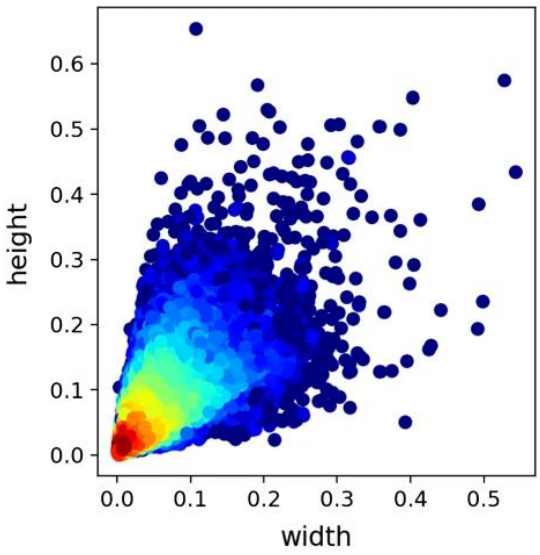


Fig. 11. Distribution map of the ratio of the length and width of the annotated target to the original image in the Visdrone2019-DET dataset.

In the Visdrone2019-DET dataset, 10 different categories of targets are marked, which contain some confusing targets, such as van and awning-tricycle. In addition, in terms of dataset division, the Visdrone2019-DET dataset is divided into a training dataset, a validation dataset and a test dataset all following the COCO dataset format [20]. Among them, the training dataset contains of 25447 image samples and their corresponding annotation files, while the verification dataset contains 1115 image samples and corresponding annotation files, and the test dataset samples 547 images. After statistical analysis, we aggregated the dataset category instances in Fig. 12. Notably, the uneven classification of the Visdrone2019-DET dataset is obvious, and the number of instances of the category "car" reached 267,470, ranking first. The number of instances of the category "awning-tricycle" is only 6648, which is at the bottom. Therefore, this makes the dataset extremely challenging.

B. Implementation Details

Our proposed algorithm was experimented on a Dell T790

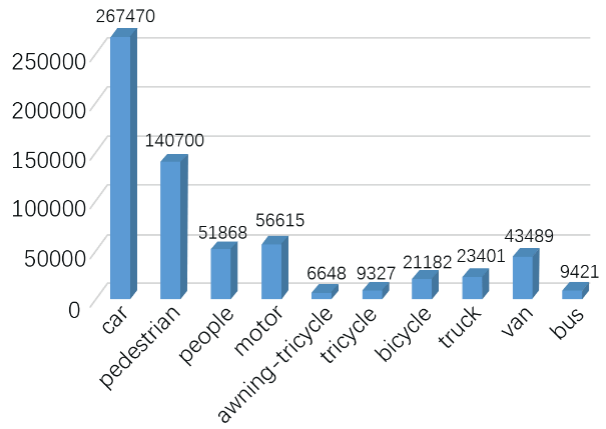


Fig. 12. Visdrone2019-DET dataset category instance total.

graphics workstation based on the pyTorch1.8.0 deep learning framework. The hardware equipment used included an Intel Xeon(R) Bronze3104 processor, two NVIDIA TITAIN XP graphics cards, 128G of server memory and two 4T solid-state drives. In terms of software, the experimental environment was set up with the Ubuntu 21.10 as the operating system, and CUDA 10.0 installed as the computing platform for parallel computing of NVIDIA graphics cards. Additionally, torchvision0.9.0, opencv4.1.2, numpy1.18.5, tensorboard2.7.0, PyYAML5.3.1, pycocotools2.0 and other Python libraries were also installed. Our improved model shares most of the Backbone with YOLOv6, and through transfer learning, many weights from YOLOv6 can be transferred to our model, saving a lot of training time. During training, start the multi-card training method and set the device to 0,1. Also, set the Batch Size to 8, the number of epochs to 200 rounds, and the number of worker processes to 4. Before training, we converted the Visdrone2019-DET dataset to the YOLO dataset format. Training usually took around 35 hours. After training, the 'best_ckpt.pt' weight and 'last_ckpt.pt' were generated in the 'runs/train' folder. The 'best_ckpt.pt' is the weight model file with the best performance during training, and the 'last_ckpt.pt' contains the weight model from the final round of training.

C. *Experimental Results and Analysis*

To accurately evaluate the good detection performance of our proposed algorithm on the Visdrone2019-DET dataset, we used Intersection-over-Union (IoU) and mean Average Precision (mAP) as our evaluation metrics. IoU represents the degree of overlap between the candidate bound and the ground truth bound, that is, the ratio of intersection to union. When the value of IoU is 1, the effect is considered ideal, indicating that the candidate bound completely overlaps with the ground truth bound. mAP is the average calculation of AP calculated for 10 categories. These 10 categories will draw a curve according to the precision rate and recall rate, and the area enclosed by the curve and the coordinate axis is the value of AP. The definitions of precision (P) and recall (R) are shown in (8) and (9):

$$P = \frac{TP}{TP + FP} \tag{8}$$

$$R = \frac{TP}{TP + FN} \tag{9}$$

TP is represented as a true positive, FP as a false positive, and FN as a false negative. In order to further optimize the results of the model, we re-evaluated the AP for different IoU value ratios and different detection target coverage areas. We taked AP^{50} and AP^{75} as evaluation metrics for under different IoU value thresholds. AP^{50} means the AP value with IoU greater than 50%, and AP^{75} means the AP value with IoU greater than 75%. AP^{S}, AP^{M}, and AP^{L} are used as the evaluation metrics for objects with different coverage areas. For objects with an area smaller than 32×32, used AP^{S} as the evaluation standard. For objects between 32×32 and 96×96 used AP^{M} as the evaluation standard, and objects with an area larger than 96×96 used AP^{L} as the evaluation standard. For the Visdrone2019-DET dataset, IoU greater than 50% is considered an effective prediction.

Similar to YOLOv5, YOLOv6 also provides multiple versions, and we list 5 different versions in Table I. The 'N' in YOLOv6-N stands for Nano, which is the fastest version with depth coefficient and width coefficient of 0.33 and 0.25, respectively. It is suitable for devices and scenarios with limited resources. YOLOv6-T and YOLOv6-S have slightly higher detection accuracy than YOLOv6-N. By increasing the number of channels in the convolutional neural network with the same depth coefficient, the parameter size and computational complexity of the network model are increased, making it suitable for detecting medium-sized objects. Comparing YOLOv6-M and YOLOv6-L, both differ from the previous versions in the structure of the Backbone and Neck. The increase in network depth and width improves the model's accuracy but also slows down the speed compared to other versions, making it suitable for detecting small-sized objects in larger and more complex scenes.

we trained the model based on the 'M' pre-trained model benchmark. Although YOLOv6-L has a deeper and wider network, its resource consumption increases the risk of overfitting, which affects the model's generalization effect. During which a verification was performed every 10 epochs, and the verification results were saved and visualized on the AP^{50} data, as shown in Fig. 13.

TABLE I
THE DIFFERENCE BETWEEN DIFFERENT VERSIONS OF YOLOV6

Method	Backbone	Neck	Head	Depth Co.	Width Co.
YOLOv6-N	EfficientRep	RepPANNeck	EffiDeHead	0.33	0.25
YOLOv6-T	EfficientRep	RepPANNeck	EffiDeHead	0.33	0.38
YOLOv6-S	EfficientRep	RepPANNeck	EffiDeHead	0.33	0.50
YOLOv6-M	CSPRepBackbone	CSPRepPANNeck	EffiDeHead	0.60	0.75
YOLOv6-L	CSPRepBackbone	CSPRepPANNeck	EffiDeHead	1.00	1.00

TABLE II
COMPARISON OF MODEL PERFORMANCE BEFORE AND AFTER IMPROVEMENT

Method	mAP(%)	AP^{50}(%)	AP^{75}(%)	AP^{S}(%)	AP^{M}(%)	AP^{L}(%)
Before	33.40	53.71	32.23	22.68	42.56	50.74
After	37.14	59.73	37.18	29.01	47.11	43.18

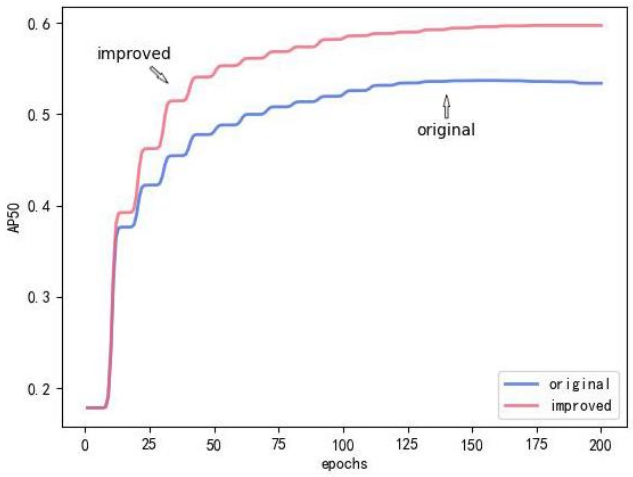


Fig. 13. Evaluation indicator data visualization.

In the figure, the accuracy of the pre-improved model reaches its peak at around 150 epochs, and the average accuracy tends to slightly decline in subsequent epochs due to overfitting. The comparison of model detection results before and after improvement is shown in Table II.

To further verify the feasibility of choosing the 'M' version as the baseline model, we compared the performance of different versions of improved YOLOv6, as shown in Table III. Although the speed of YOLOv6-M is lower than that of YOLOv6-T and YOLOv6-S, its accuracy is much higher than the two. Because the Visdrone2019-DET dataset contains numerous small-sized objects, resulting in poor detection performance. Comparing YOLOv6-M and YOLOv6-L, the mean Average Precision differs by only 0.13%, but the speed gap is significant. Therefore, YOLOv6-M has a better balance between accuracy and speed, making it more suitable for detecting scenes with numerous small-sized objects.

TABLE III
PERFORMANCE COMPARISON BETWEEN DIFFERENT VERSIONS OF YOLOV6 AFTER IMPROVEMENT

Method	mAP(%)	AP^{50} (%)	FPS
YOLOv6-T	32.71	52.54	19.6
YOLOv6-S	33.68	54.16	18.3
YOLOv6-M	37.14	59.73	15.7
YOLOv6-L	37.27	60.47	6.8

Fig. 14 shows the comparison results of the Visdrone test dataset before and after improvement. Fig. 14(a) is the visualization result before improvement, and Fig. 14(b) is the visualization result after improvement. Bounding boxes of different colors are used to represent different categories. In the high-density scene, the improved model detection results have been greatly improved, as shown in the first row of Fig. 14, it better recognizes small targets in the distance, and identifies the target category in the near distance more accurately. The second row shows that our improved model can still accurately detect small objects in brighter light source scenes. The third row shows that our proposed model still achieves superior performance in scenes with darker light sources.

To validate the effectiveness of our approach, we compared and analyzed the detection performance on the Visdrone dataset with YOLO series algorithms, Anchor-free detection series algorithms, Transformer series algorithms, etc., as shown in Table IV.

The comparison found that among the YOLO series algorithms, YOLOv5-L's AP^{50} is the highest, reaching 60.86%. Ranked second is YOLOv5-M, reaching 59.57%. Comparing the two, AP^{50} of YOLOv5-L is less than 2% higher than that of YOLOv5-M, but the detection speed is far behind that of the latter, and the real-time performance is poor. CornerNet [21] and YOLOX [14] are the more popular Anchor-free detection series algorithms. From the performance point of view, the Anchor-free detector has low regression accuracy and poor effect. The backbone of MobileViT [22] uses the Transformer structure globally, which does not have an advantage in small target detection.

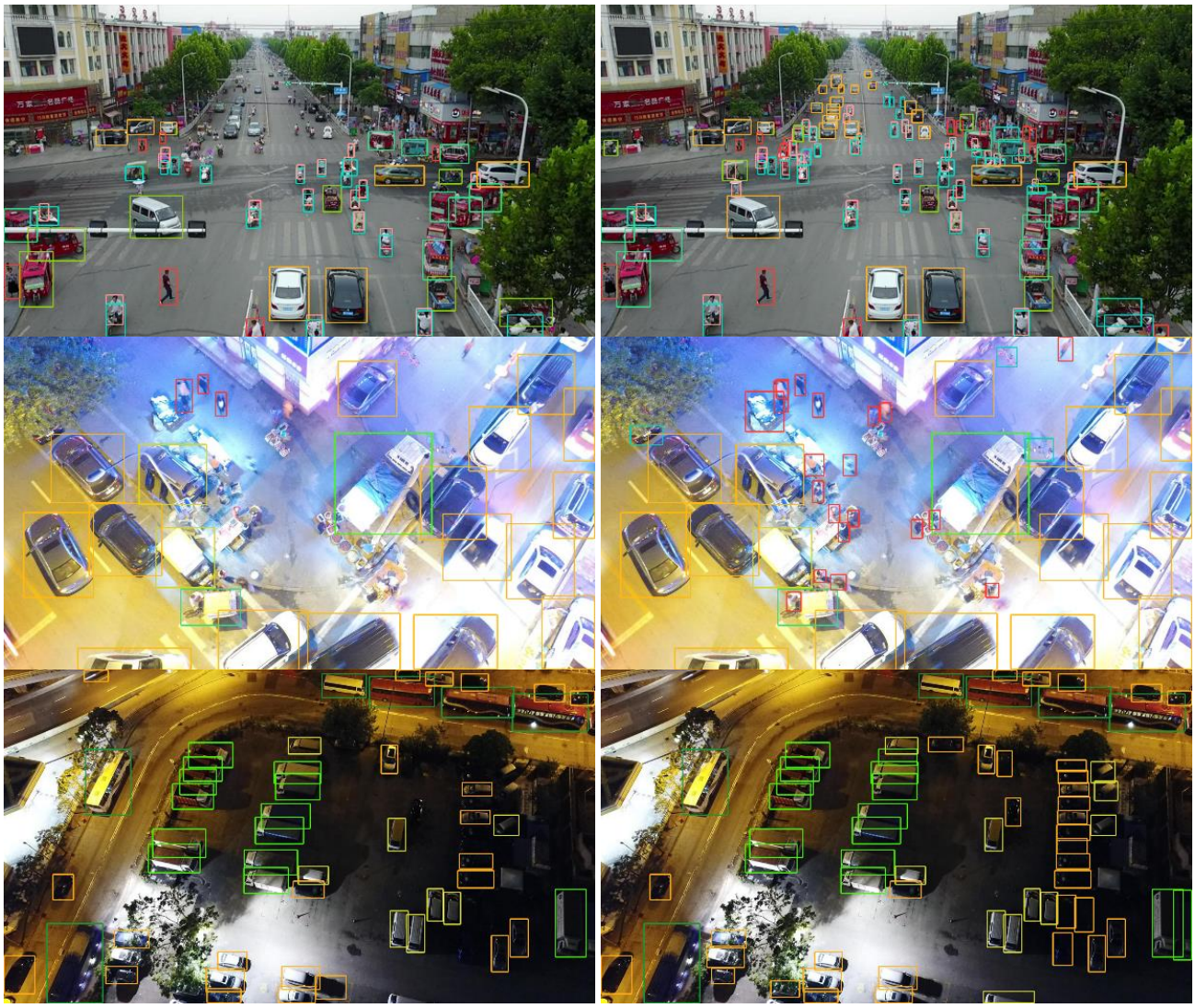
D. Ablation Experiments

Based on the original model, this paper proposes three improvement measures, and the impact of each improvement is listed in Table V.

TABLE V
THE IMPACT OF EACH IMPROVEMENT ON THE MODEL DETECTION PERFORMANCE OF

Method	MDBA	FaPN	TR-Head	mAP(%)	AP^{50} (%)
YoloV6				33.40	53.71
Ours	√			34.23	55.04
Ours		√		34.43	55.38
Ours			√	35.15	56.53
Ours	√	√		35.28	56.75
Ours	√		√	36.11	58.06
Ours		√	√	36.32	58.40
Ours	√	√	√	37.14	59.73

MDBA is a mixed data augmentation of background samples method proposed in this paper. When only this improvement is added to the model, the detection accuracy increases by 1.33%, which shows that the MDBA data augmentation method can better help the model learn which targets need to be detected, which ones are background samples. The model can maintain good detection performance in complex and changing conditions, such as rainy and snowy weather, haze weather, cloudy weather, too dark environment, too bright environment, etc. In particular, data augmentation is a model training technique oriented towards optimizing the dataset. When the computational complexity is guaranteed to remain unchanged, FLOPs and



(a) Before improvement (b) After improvement

Fig. 14. Visdrone test set visualization results before and after model improvement.

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT ALGORITHMS ON THE VISDRONE DATASET

Method	mAP(%)	AP^{50}(%)	AP^{75}(%)	AP^{S}(%)	AP^{M}(%)	AP^{L}(%)	FLOPs	FPS
YOLOv3	31.57	48.31	31.32	20.94	40.63	46.76	154.9G	27.8
YOLOv4	32.88	51.37	32.54	21.37	41.65	48.78	119.4G	23.4
YOLOv5-S	31.93	54.83	31.75	22.08	43.75	49.51	16.5G	10.6
YOLOv5-M	36.52	59.57	37.21	28.56	48.23	52.57	50.4G	9.8
YOLOv5-L	37.08	60.86	38.12	29.22	49.36	53.23	98.6G	4.8
CornerNet	30.82	52.08	29.74	21.15	37.83	43.54	37.7G	6.8
YOLOX-S	31.47	53.53	29.74	22.56	41.58	48.52	41.7G	5.1
MobileViT	33.43	55.45	33.76	24.92	44.28	41.87	-	13.7
TPH-YOLOv5	36.45	58.44	36.49	28.48	46.09	42.41	68.3G	18.3
Ours	37.14	59.73	37.18	29.01	47.11	43.18	76.4G	15.7

Parameters will not increase, which also makes the FPS unchanged from the original model.

When FaPN is added to the Neck structure of YOLOv6 alone, the detection accuracy is 55.38%, which is 1.67% higher than that of YOLOv6. This method of feature alignment has certain effectiveness in the Visdrone dataset containing many small targets. It further learns the offset of each sampling position, thereby preventing the loss of small target features during the upsampling process. Adding a feature selection module with an attention mechanism to the skip connection of the pyramid structure can better emphasize the details before upsampling.

When only the TR-Head improvement is added to the model, the detection accuracy increases by nearly 3 percentage points compared to the original model. Among the three improvement measures proposed in this paper, the single improvement has the highest efficiency. By applying the Transformer Encoder Block to different levels of the upsampling process makes up for the lack of a single input size and maximizes its effectiveness. At the same time, it also

plays a certain role in the detection of high-density objects and small target objects.

V. Conclusion

In this study, we propose a UAV object detection algorithm based on improved YOLOv6, which improves the ability to detect high-density, small target objects in UAV aerial images. First, we designed a data augmentation method in which background samples are added to the training. This enables the model to better distinguish between true positives and false positives without affecting the computational complexity of the original model. Secondly, to prevent loss of feature information caused by small target features during the upsampling stage, we added FaPN to the Rep-PAN structure, making the Neck of the detection network more powerful. Finally, we introduced Transformer into CNN, and used Transformer Encoder Block to form Transformer Prediction Head, which replaced the prediction head of the original model. Our proposed algorithm achieved an accuracy of 59.73% on the Visdrone dataset, which is 6 percentage points higher than before the improvements. We also compared with different algorithms, our proposed algorithm achieved better detection accuracy and detection speed than other algorithms. The detection performance of our improved model is relatively impressive. In the future, we still need to pay attention to the impact of recalling on detection accuracy and the actual test deployment on UAV parallel equipment.

References

[1] Shahbazi M, Théau J, Ménard P. Recent applications of unmanned aerial imagery in natural resource management[J]. GIScience & Remote Sensing, 2014, 51(4): 339-365.

[2] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.

[3] Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. arXiv 2016[J]. arXiv preprint arXiv:1612.08242, 2016, 394.

[4] Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[5] Bochkovskiy A, Wang C Y, Liao H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[6] Attari N, Ofli F, Awad M, et al. Nazr-CNN: object detection and fine-grained classification in crowdsourced UAV images[C]//IEEE International Conference on Data Science and Advanced Analytics (DSAA). 2016.

[7] Majid Azimi S. Shuffledet: real-time vehicle detection network in on-board embedded uav imagery[C]//Proceedings of the European Conference on Computer Vision (ECCV) Workshops. 2018: 0-0.

[8] Duan K, Bai S, Xie L, et al. Centernet: Keypoint triplets for object detection[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 6569-6578.

[9] Carion N, Massa F, Synnaeve G, et al. End-to-end object detection with transformers[C]//European conference on computer vision. Springer, Cham, 2020: 213-229.

[10] Zhu X, Lyu S, Wang X, et al. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 2778-2788.

[11] Li C, Li L, Jiang H, et al. YOLOv6: A single-stage object detection framework for industrial applications[J]. arXiv preprint arXiv:2209.02976, 2022.

[12] Ding X, Zhang X, Ma N, et al. Repvgg: Making vgg-style convnets great again[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 13733-13742.

[13] Tian Z, Shen C, Chen H, et al. FCOS: Fully convolutional one-stage object detection[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 9627-9636.

[14] Ge Z, Liu S, Wang F, et al. YOLOX: Exceeding yolo series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.

[15] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929, 2020.

[16] Dong Y, Cordonnier J B, Loukas A. Attention is not all you need: Pure attention loses rank doubly exponentially with depth[C]//International Conference on Machine Learning. PMLR, 2021: 2793-2803.

[17] Zhang H, Cisse M, Dauphin Y N, et al. mixup: Beyond empirical risk minimization[J]. arXiv preprint arXiv:1710.09412, 2017.

[18] Huang S, Lu Z, Cheng R, et al. FaPN: Feature-aligned pyramid network for dense image prediction[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 864-873.

[19] Du D, Zhu P, Wen L, et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results[C]//Proceedings of the IEEE/CVF international conference on computer vision workshops. 2019: 0-0.

[20] Lin T Y, Maire M, Belongie S, et al. Microsoft coco: Common objects in context[C]//European conference on computer vision. Springer, Cham, 2014: 740-755.

[21] Law H, Deng J. Cornernet: Detecting objects as paired keypoints[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 734-750.

[22] Mehta S, Rastegari M. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer[J]. arXiv preprint arXiv:2110.02178, 2021.