electronics

Article

An Improved Lightweight Real-Time Detection Algorithm Based on the Edge Computing Platform for UAV Images

Lijia Cao [1,2,3,4,*], Pinde Song [1], Yongchao Wang [5], Yang Yang [1] and Baoyu Peng [4]

[1] School of Automation & Information Engineering, Sichuan University of Science & Engineering, Yibin 644000, China; 321085404206@stu.suse.edu.cn (P.S.)
[2] Artificial Intelligence Key Laboratory of Sichuan Province, Yibin 644000, China
[3] Sichuan Province University Key Laboratory of Bridge Non-Destruction Detecting and Engineering Computing, Yibin 644000, China
[4] School of Computing Science and Engineering, Sichuan University of Science & Engineering, Yibin 644000, China
[5] Chair of Automatic Control Engineering, Technical University of Munich, 80333 Munich, Germany
* Correspondence: caolj@suse.edu.cn

Abstract: Unmanned aerial vehicle (UAV) image detection algorithms are critical in performing military countermeasures and disaster search and rescue. The state-of-the-art object detection algorithm known as you only look once (YOLO) is widely used for detecting UAV images. However, it faces challenges such as high floating-point operations (FLOPs), redundant parameters, slow inference speed, and poor performance in detecting small objects. To address the above issues, an improved, lightweight, real-time detection algorithm was proposed based on the edge computing platform for UAV images. In the presented method, MobileNetV3 was used as the YOLOv5 backbone network to reduce the numbers of parameters and FLOPs. To enhance the feature extraction ability of MobileNetV3, the efficient channel attention (ECA) attention mechanism was introduced into MobileNetV3. Furthermore, in order to improve the detection ability for small objects, an extra prediction head was introduced into the neck structure, and two kinds of neck structures with different parameter scales were designed to meet the requirements of different embedded devices. Finally, the FocalEIoU loss function was introduced into YOLOv5 to accelerate bounding box regression and improve the localization accuracy of the algorithm. To validate the performance of the proposed improved algorithm, we compared our algorithm with other algorithms in the VisDrone-Det2021 dataset. The results showed that compared with YOLOv5s, MELF-YOLOv5-S achieved a 51.4% reduction in the number of parameters and a 38.6% decrease in the number of FLOPs. MELF-YOLOv5-L had 87.4% and 47.4% fewer parameters and FLOPs, respectively, and achieved higher detection accuracy than YOLOv5l.

Keywords: lightweight; FocalEIoU; UAV image; attention mechanism; embedded device

check for updates

Citation: Cao, L.; Song, P.; Wang, Y.; Yang, Y.; Peng, B. An Improved Lightweight Real-Time Detection Algorithm Based on the Edge Computing Platform for UAV Images. *Electronics* **2023**, *12*, 2274. https://doi.org/10.3390/electronics12102274

Received: 15 April 2023
Revised: 12 May 2023
Accepted: 16 May 2023
Published: 18 May 2023

1. Introduction

1.1. Background

In recent years, two methods have emerged for unmanned aerial vehicle (UAV) image detection: traditional feature description and deep learning-based methods. Both of these methods have shown excellent results in performing the detection task. Based on the aforementioned methods, many researchers have conducted extensive research on UAV image detection.

In the past decade, traditional manual feature methods were widely used to detect UAV images, such as the Viola–Jones (VJ) detector [1], the histogram of oriented gradients (HOG) [2], and the deformable part-based model (DPM) detector [3]. Wang et al. proposed a method based on face detection (FD) and HOG to detect drones [4]. Xu et al. proposed a new hybrid vehicle detection scheme using a VJ detector and a support vector machine

(SVM) classifier with a HOG feature method [5]. Jiang et al. proposed a method based on DPM to detect transmission towers [6]. However, the above methods rely heavily on the rules of artificial design and can only learn features at the pixel level. In addition, these methods have some limitations, such as poor generalization ability, poor detection ability, and complex design processes.

With the development of deep learning, convolutional neural networks (CNNs) have shown excellent feature extraction abilities and backbone networks such as AlexNet [7], VGG [8], GoogleNet [9], and ResNet [10] have exhibited satisfactory performance in various midstream and downstream tasks. Deep learning-based object detection algorithms can be categorized into two types: region recommendation-based and regression-based.

Region Recommendation-Based Algorithms: In 2014, Ross et al. proposed a detection algorithm [11] founded on AlexNet. The algorithm used Selective Search [12] to generate region proposals, which were then individually fed into the neural network to perform detection and classification. Although the region-based convolution neural network (RCNN) applied neural networks to target detection for the first time, it still had many limitations: (1) thousands of region proposals mostly overlapped each other, and the overlapping parts were repeatedly extracted as features many times; (2) scaling the region proposals directly to a fixed size destroyed the aspect ratio of the object and could lead to loss of local detail in the object; and (3) excessive training time and complex training process. Ross et al. proposed an improved algorithm [13], which addressed some of the limitations of RCNN, such as the slow training and testing time. Fast-RCNN also used Selective Search to generate region proposals, but instead of applying the neural network to each proposal, it applied the network to the entire image and used a region of interest (ROI) pooling layer to map each region proposal to the corresponding feature map location. Finally, Fast-RCNN used Softmax to classify and regress the proposal positions to complete the detection. Despite being a significant advancement over RCNN, Fast-RCNN essentially used Selective Search to generate region proposals, which significantly slowed down the network's detection speed. On the basis of Fast-RCNN, Faster-RCNN [14] was proposed by He et al. in 2016. The algorithm used a region proposal network (RPN) to automatically generate region proposals for end-to-end training, while also optimizing the proposal extraction technique. As a result, Faster-RCNN made tremendous progress in terms of detection speed and accuracy. However, these algorithms, such as RCNN, Fast-RCNN, and Faster-RCNN, have difficulty meeting the requirements of real-time detection.

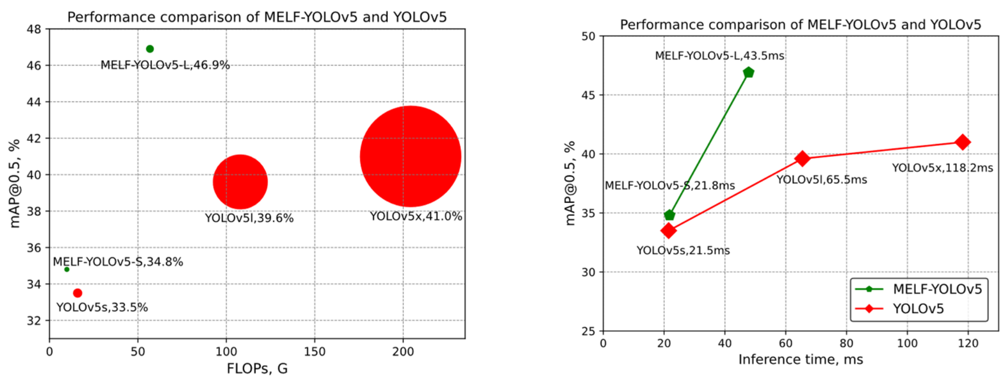
Regression-Based Algorithms: Although RCNN series algorithms had satisfactory detection accuracy, it was difficult to meet real-time requirements for complex scenes. In 2015, Joseph et al. proposed a one-stage detection algorithm, which was based on regression [15]. The algorithm fed the image into the network to obtain the regression parameters, classification, and confidence. End-to-end detection was achieved by the you only look once (YOLO) algorithm, which significantly sped up detection. However, it was easy for YOLO to miss detection when the target object was much denser, and the detection accuracy of this algorithm was significantly lower than that of other two-stage detection algorithms. In 2016, Joseph et al. proposed the YOLOv2 algorithm [16], which increased the regression parameters of each grid. This algorithm significantly improved the detection ability. On the basis of YOLOv2, YOLOv3 was proposed by Joseph et al. [17]. The feature pyramid network (FPN) structure was introduced in YOLOv3, which achieved multi-scale detection. In 2020, Alexey et al. proposed YOLOv4 [18]. The algorithm balanced accuracy and speed by using methods such as lightweight cross stage partial (CSP) [19], Mish activation function, better loss function, and path aggregation network (PANet). In 2021, Glenn et al. released YOLOv5, based on the YOLO family, which combined common optimization strategies with more a complex network architecture design. In addition, the lightweight modification CSP (C3) module was used as the backbone network of YOLOv5. YOLOv5 was a highly effective, one-stage object detection algorithm that could make fast and accurate predictions on a single GPU.

In recent years, YOLOv5, the fastest algorithm in the YOLO series, has been widely used in UAV image object detection tasks. Wang et al. proposed a YOLO-D model based on a dual-attention mechanism for the detection of offshore UAV images [20]. Li et al. proposed an improved attention module and added a small detection head to capture small objects, thus enhancing the detection ability for small objects [21]. Based on YOLOv5, Zhu et al. replaced the prediction head with transformer encoder blocks and added a small object detection head for UAV image detection [22]. However, when the TPH-YOLOv5 algorithm was deployed to edge computing devices, it could not achieve real-time detection. On the basis of YOLOv5, Jung et al. improved the backbone and activation function for complex environments. A novel object detection algorithm was proposed by Li et al. for drone cruising in large-scale maritime scenarios [23], which reduced the computational cost by replacing the convolution operations with simpler linear transformations. Li et al. designed a lightweight backbone network to reduce the computational cost [24]. The algorithm achieved real-time detection in the Jetson AGX edge computing platform. Dong et al. optimized the overall structure of the model based on YOLOv5s by introducing channels and spatial attention mechanisms to the backbone network, while replacing the 1×1 convolution and C3 modules in the neck structure with the Ghost and C3Ghost modules. However, due to the algorithm only having three detection heads, it still had some limitations in the accuracy of small object detection. Cheng et al. proposed a real-time object detection algorithm [25], but the number of parameters was over ten million and the FPS only had a few frames in Jetson Nano.

1.2. Contributions

To balance detection accuracy and inference speed, we propose a lightweight real-time detection algorithm in this paper. Figure 1 presents a performance comparison between the MELF-YOLOv5 and YOLOv5 algorithms. The main contributions of this paper can be summarized as follows:

(1) We compare the performance of the depthwise separable convolution module and the C3 module in terms of parameter compression via calculations.
(2) The lightweight MobileNetv3 and ECA attention mechanism are used to compress the backbone network of YOLOv5.
(3) A prediction head is added improve the detection ability of the model.
(4) The FocalEIoU loss function is introduced into YOLOv5 to improve the localization accuracy.
(5) Two kinds of neck structures are designed to meet the needs of different embedded devices.



(**a**) (**b**)

Figure 1. Comparisons between YOLOv5 and MELF-YOLOv5. (**a**) Shows the numbers of parameters and FLOPs and the mAP@0.5 of MELF-YOLOv5 and YOLOv5, where the area of the circle represents the number of parameters; (**b**) shows the inference time and mAP@0.5 of MELF-YOLOv5 and YOLOv5.

1.3. Organization

The paper is structured as follows: Section 2 provides an introduction to the principle and structure of YOLOv5 and MobileNetV3. In Section 3, we propose the MELF-YOLOv5 algorithm. Section 4 presents a discussion of the experimental results. Finally, the conclusion is presented in Section 5.

2. Methods

YOLOv5 is an efficient, one-stage target detection algorithm that contains five models with different parameter scales, such as YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. YOLOv5n has the highest inference efficiency and YOLOv5x has the best detection accuracy among these models. In addition, these models are all composed of the same structures, including the backbone, FPN [26], PAN [27], and prediction heads. The detailed description of each module of the YOLOv5 model is as follows.

The YOLOv5 backbone network includes several modules, such as spatial pyramid pooling fast (SPPF) Conv, and C3 module. The SPPF is an improvement based on spatial pyramid pooling (SPP) [28], which can produce fixed-size output. Conv is a convolution unit, which is composed of a convolution layer, batch normalization (BN) layer, and activation function. As shown in Figure 2, the C3 module is composed of Conv, residual, and serval CBS modules.

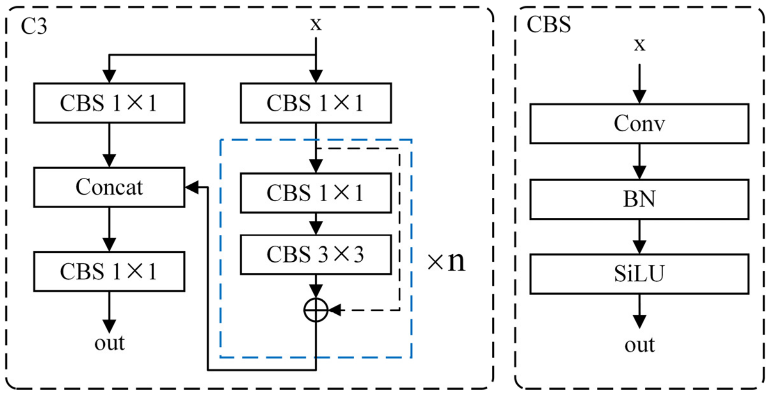


Figure 2. The structure of C3 module.

In the FPN, top semantic information is fused into lower level layers to enhance their expressiveness. Additionally, the targets of different scales can be assigned to different layers for prediction.

The PANet transfers the positional feature from lower feature maps to upper feature maps and combines the semantic information from the FPN to achieve both classification and position.

There are three prediction head in YOLOv5, and each head outputs confidence, classification, and the positional information of the bounding box for three anchors. The confidence and classification loss is BCE-WithLogitsLoss, which is essentially no different from biniary crossentropy loss (BCELoss) except the value of BCELoss is processed by sigmoid. The mathematical expression of BCELoss is:

$$loss(p,y) = -\frac{1}{n}\sum_{i}\left[y^i \log\left(p^i\right) + \left(1-y^i\right)\log\left(1-p^i\right)\right] \tag{1}$$

where y^i shows whether or not the category is predicted ($y^i = 0$ refers to a non-predicted category and $y^i = 1$ refers to a predicted category), and p^i is the probability that the algorithm predicts the output of a category from the head.

The bounding box loss function is complete intersection over union (CIoU) [29] in YOLOv5. It can be computed as follows:

$$\begin{aligned} CIOU_{loss} &= 1 - IOU + \frac{\rho^2\left(b^p, b^{gt}\right)}{c^2} + \alpha\nu \\ \alpha &= \frac{\nu}{(1-IOU)+\nu} \\ \nu &= \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2 \end{aligned} \tag{2}$$

where α denotes the weight coefficient, ν is the aspect ratio similarity between the target and predicted, c denotes the diagonal distance between the ground truth and prediction boxes, ρ is the Euclidean distance, b^{gt} is the coordinate information of the ground truth box, b^p is the coordinate information of the prediction box, and IOU is the intersection ratio of the ground truth and prediction boxes.

2.1. MobileNetV3 Backbone Network

On the basis of MobileNetV1 (V1) [30] and MobileNetV2 (V2) [31], Howard et al. proposed MobileNetV3 (V3) [32]. The Hardwish activation function was proposed and used in the V3 backbone network. In addition, the squeeze and excitation (SE) [33] channel attention mechanism was also introduced into the depthwise separable convolution module.

The SE channel attention mechanism was introduced in the depthwise separable convolution module to improve the feature extraction ability of V3. As shown in Figure 3, the SE channel attention mechanism consists of a pooling layer, two fully connected layers, and a hard sigmoid activation function. The SE module focus certain information in the channel and generates a weight for each feature channel, and then these weights are multiplied with the input feature maps element-wise to obtain the final feature maps.

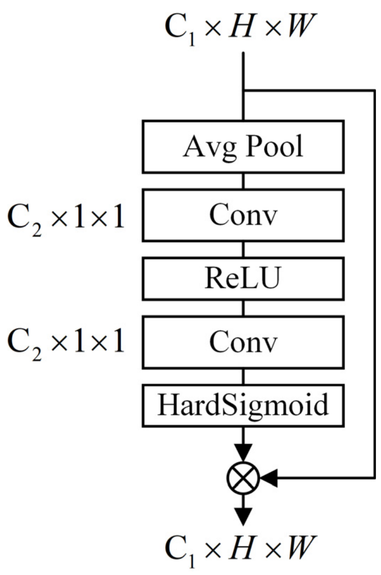


Figure 3. SE attention module.

Figure 4 illustrates the implementation of the depthwise separable module. The high-dimensional feature maps are mapped to the lower dimension, which reduces the number of parameters while enhancing the information correlation in the channels. Secondly, depthwise convolution is used to extract the features. Then, the outputs are mapped to higher dimensions to enhance feature interaction in the channels. Finally, the last feature maps are processed by the SE channel attention mechanism, which allows the network to focus more on specific feature information in each channel.

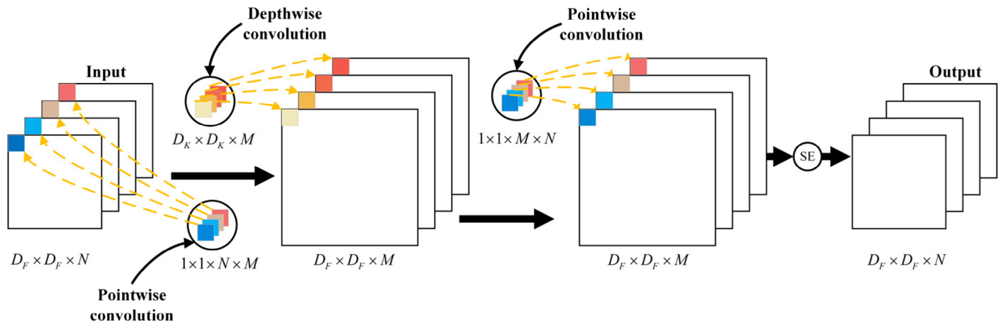

Depthwise convolution
Pointwise convolution
Input
Output
$D_K \times D_K \times M$
$1 \times 1 \times M \times N$
SE
$D_F \times D_F \times N$
$1 \times 1 \times N \times M$
$D_F \times D_F \times M$
$D_F \times D_F \times M$
$D_F \times D_F \times N$
Pointwise convolution

Figure 4. Depthwise separable convolution module based on SE.

In Figure 4, D_F is the output size of a feature maps, D_K is the size of the convolution kernel, M is the number of filters, and N denotes the number of channels and filters.

When comparing the ordinary convolution module with the depthwise separable convolution module, the FLOPs of the depthwise separable convolution module can be calculated as:

$$P_d = D_F \times D_F \times N \times M + D_F \times D_F \times D_K \times D_K \times M + D_F \times D_F \times M \times N \tag{3}$$

As shown in Figure 3, the FLOPs of the SE channel attention mechanism can be computed as:

$$Q_{SE} = D_F \times D_F \times M + M \times \frac{M}{4} + \frac{M}{4} \times M \tag{4}$$

Under the same parameter conditions, we can calculate the FLOPs of ordinary convolution as:

$$P_o = D_K \times D_K \times N \times D_F \times D_F \times N \tag{5}$$

In accordance with Equations (3)–(5), we have:

$$\frac{P_{V3}}{P_o} = \frac{Q_{SE} + P_d}{P_o} = \frac{M}{N^2} + \frac{2M}{N \times D_K^2} + \frac{M}{N^2 + D_K^2} + \frac{M^2}{2N^2 \times D_K^2 \times D_F^2} \tag{6}$$

where P_{V3} is the FLOPs of the depthwise separable convolution module, which is based on the SE channel attention mechanism.

Because the number of filters M is much smaller than $N^2 \times D_K^2$, and M^2 is also much smaller than $2N^2 \times D_K^2 \times D_F^2$, we have:

$$\frac{M}{N^2 \times D_K^2} + \frac{M^2}{2N^2 \times D_K^2 \times D_F^2} \approx 0 \tag{7}$$

Then,

$$\frac{P_{V3}}{P_o} \approx \frac{M}{N^2} + \frac{2M}{N \times D_K^2} \tag{8}$$

In fact, the size of the filters is often smaller than the number of channels. In terms of that, we scaled up the channels from M to N in order to simplify the calculation. The size of the filters was set to 3. Thus, we have:

$$\frac{P_{V3}}{P_o} \approx \frac{M}{N^2} + \frac{M}{N} \times \frac{2}{D_K^2} < \frac{1}{N} + \frac{2}{D_K^2} \approx \frac{2}{D_K^2} = \frac{2}{9} \tag{9}$$

From Equation (9), it can be seen that the number of FLOPs of the ordinary convolution is approximately 4.5 times that of the depthwise separable convolution module with SE.

2.2. ECA Attention Mechanism

ECA is an efficient attention mechanism [34]. This module, bringing significant performance gains by using a small number of parameters, overcomes the contradiction between performance and complexity.

As shown in Figure 5, the ECA is a lightweight channel attention mechanism, given a feature map $F \in R^{M \times D_F \times D_F}$ as input. Firstly, the input is dealt to a vector by average pooling. Then, a convolution operation is used to extract channel information from the vector. Finally, the sigmoid activation function is used for the vector. The overall attention process can be summarized as:

$$F' = \sigma\left(F_{avg}(F) \otimes W_1\right) \tag{10}$$

where $W_1 \in R^{1 \times 1 \times 3}$ denotes the weights of channel attention, $\otimes$ is a convolution operation, F_{avg} denotes average pooling operation, and σ is the sigmoid activation function.

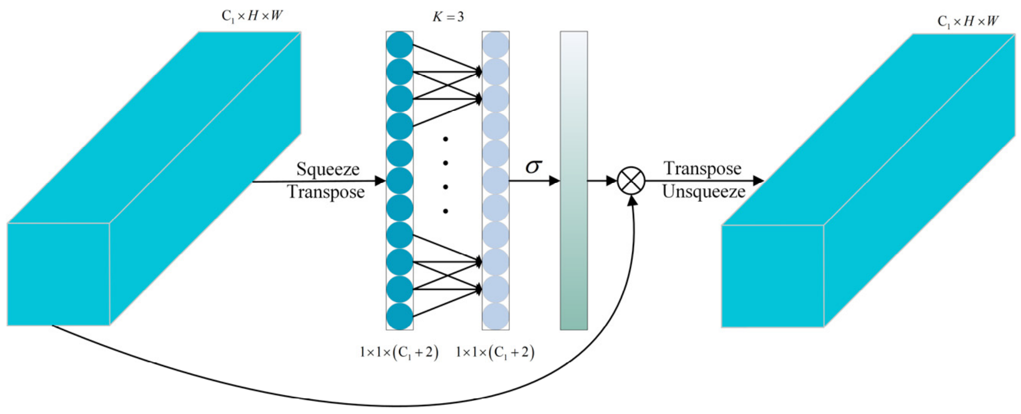


Figure 5. ECA channel attention mechanism.

3. YOLOv5 Algorithm Improvement

An object detection method based on YOLOv5 is proposed to balance the detection accuracy and inference speed of UAV image detection in this paper. The proposed method was designed to be lightweight and achieve real-time detection. Figure 6 shows the improved YOLOv5 structure named MELF-YOLOv5. In the structure, as shown in Figure 6g, MobileNetV3 and the ECA module were introduced into the backbone network of YOLOv5. Additionally, in Figure 6h, a prediction head was added in the shallow layer, which enhanced the ability of model to focus on small objects. FocalEIoU [35] was used as the box regression of loss function. Finally, to meet the needs of different embedded devices, two kinds of convolution blocks were introduced into the neck structure, as shown in Figure 6e,f.

3.1. Improvement of the Backbone Network

In order to reduce the numbers of parameters and FLOPs inYOLOv5, the V3 backbone network was introduced into YOLOv5, while the SE channel attention mechanism was replaced by the ECA module to improve the feature extraction ability and reduce the inference time.

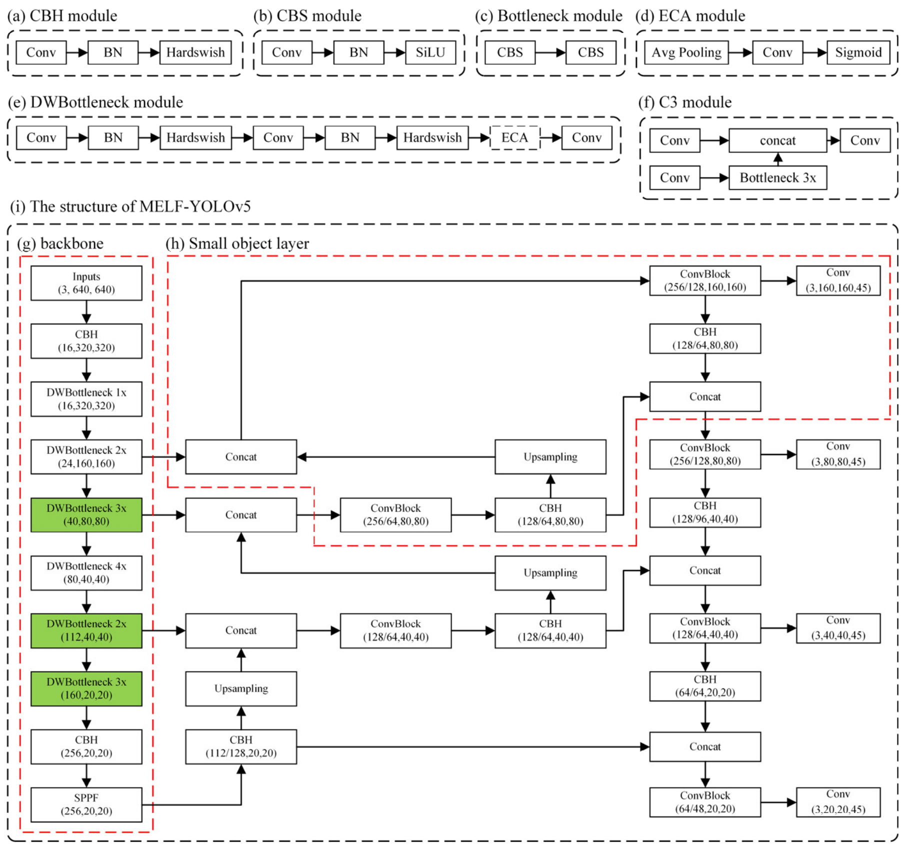


Figure 6. The structure of MELF-YOLOv5.

MobileNetV3 is a lightweight backbone network with fewer parameters and FLOPs than the C3 module. As shown in Figure 4, the FLOPs of the depthwise separable convolution module can be calculated as Equations (3) and (4). Under the same conditions, given a feature map $D_F \times D_F \times N$ as input, the FLOPs of the C3 module are computed as:

$$\begin{aligned} P_{C3} &= N \times \tfrac{N}{2} \times D_F \times D_F + N \times \tfrac{N}{2} \times D_F \times D_F + N \times N \times D_F \times D_F \\ &\quad +3\left(\tfrac{N}{2} \times \tfrac{N}{2} \times D_F \times D_F + D_K \times D_F \times \tfrac{N}{2} \times \tfrac{N}{2} \times D_F \times D_F\right) \\ &= 2N^2 \times D_F^2 + 0.75N^2 \times D_K^2 \times D_F^2 + 0.75N^2 \times D_F^2 \\ &= N^2 \times D_F^2 \times \left(2.75 + 0.75D_K^2\right) \end{aligned} \tag{11}$$

To simplify the calculation, we scaled up the channels from M to N, and the size of filter was set to 3. Additionally, we also ignored the impact of SE on the FLOPs. Thus, we have:

$$\frac{P_{V3}}{P_{C3}} = \frac{D_F \times D_F \times N \times M + D_F \times D_F \times D_K \times D_K \times M + D_F \times D_F \times M \times N}{D_F^2 \times N^2 \times \left(2.75 + 0.75D_K^2\right)} \tag{12}$$

$$\frac{P_{V3}}{P_{C3}} = \frac{1}{2.75 + 0.75D_K^2} \times \left(2 + \frac{D_K^2}{N}\right) = \frac{1}{9.5} \times \left(2 + \frac{9}{N}\right) < \frac{2.5}{9.5} = \frac{5}{19} \tag{13}$$

From Equation (13), the number of FLOPs of the C3 module can be computed as 3.8 times that of the depthwise separable convolution module. Similarly, the calculation of parameters can be approximated to 3.8. Obviously, the depthwise separable convolution module can reduce the computational cost and compress the parameters of the model.

3.2. Improvement of the Prediction Head

Detecting small objects is a challenging task for object detection algorithms as they have fewer pixels during the convolution process. As the network layers become deeper, it becomes easier for the features of large objects to be retained, while small objects are more likely to be ignored. To address this issue, we added a prediction layer for small objects (cf., Figure 7).

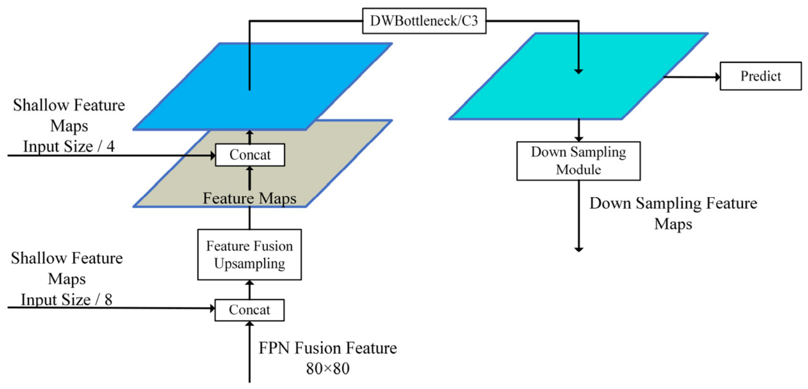

DWBottleneck/C3
Predict
Shallow Feature Maps Input Size / 4
Concat
Feature Maps
Down Sampling Module
Down Sampling Feature Maps
Feature Fusion Upsampling
Shallow Feature Maps Input Size / 8
Concat
FPN Fusion Feature 80×80

Figure 7. The prediction layer for small objects.

In the prediction layer for small objects, in order to improve the detection ability for small objects, we added feature maps with quadruple downsampling and a smaller receptive field. The resulting fused features were then input into the PAN module to enhance the ability of network to determine the objects' positions.

3.3. Loss Function Improvement

The regression loss function used by YOLOv5 is CIoU. The loss function evaluates the difference between the prediction and ground truth boxes based on the overlapping area, center point distance, and aspect ratio. However, the anti-trigonometric function used in CIoU results in an increase in computational cost and training time. In addition, the CIoU loss function ignores the differences in width and height between the prediction and ground truth boxes. Therefore, in this paper, FocalEIoU was used as the regression loss function. It can be expressed as:

$$L_{EIoU} = 1 - IoU + \frac{\rho\left(b, b^{gt}\right)}{c^2} + \frac{\rho\left(w, w^{gt}\right)}{c_w^2} + \frac{\rho\left(h, h^{gt}\right)}{c_h^2} \tag{14}$$

$$L_{FocalEIoU} = IoU^{\gamma} L_{EIoU} \tag{15}$$

where ρ denotes the Euclidean distance; c denotes the diagonal distance enclosing box covering the two boxes; c_w and c_h denote the width and height of the smallest enclosing box covering the two boxes, respectively; b^{gt} and b denote the central x-coordinates of the ground truth and prediction boxes, respectively; w and w^{gt} denote the widths of the ground truth and prediction boxes, respectively; h and h^{gt} denote the heights of the ground truth and prediction boxes respectively; and γ denotes a hyperparameter.

Compared with CIoU, FocalEIoU addresses the problem of incorrectly enlarging the side length between the prediction and ground truth boxes, resulting in better box regression than CIoU.

4. Experiments

4.1. Experimental Introduction

4.1.1. Experimental Environment

The training environment for the experiments was the Ubuntu 18.04 operating system, and the YOLOv5 algorithm was implemented using PyTorch. The training server had a hardware configuration of dual Nvidia RTX8000 and Intel Xeon Gold 6254 processors. The Nvidia Jetson AGX Xavier was used to test the performance on an embedded platform. Furthermore, the image input size for all experimental models was 640 $\times$ 640 pixels.

4.1.2. Experimental Dataset

To evaluate the proposed method, the VisDrone-DET2021 [36] dataset, including 10 categories, was used for the experiments. The training and validation sets had 6471 and 549 images, respectively. Because the format of this dataset did not match the one required by the YOLOv5 algorithm, we converted the dataset in terms of position, size, and category. Representative sample data from the perspective of UAVs are shown in Figure 8.

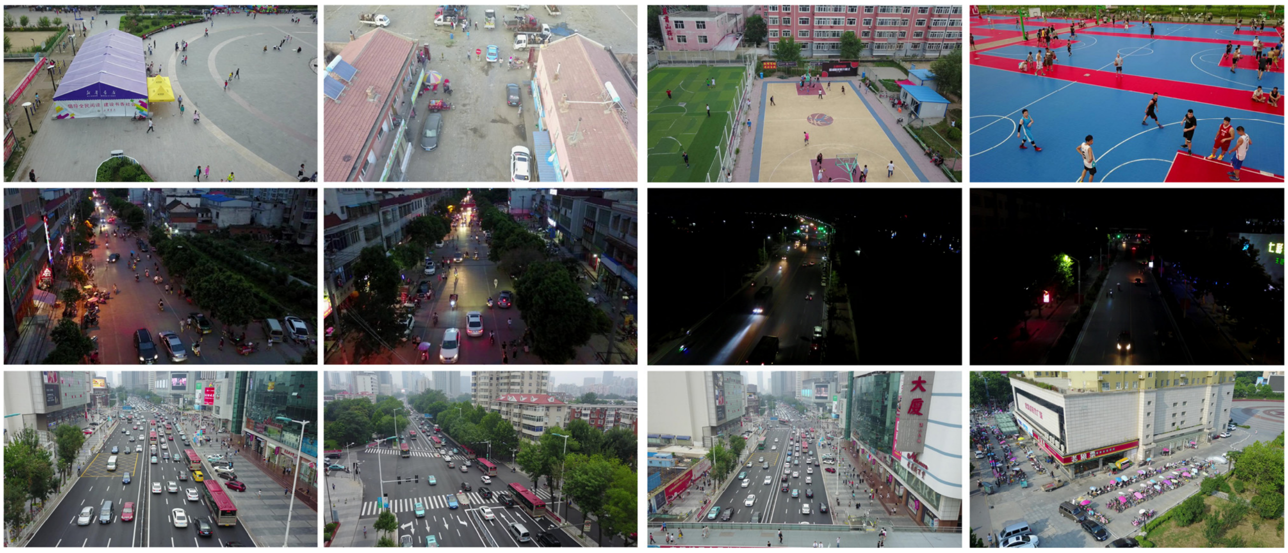


Figure 8. Representative sample data.

4.1.3. Evaluating Indicators

To verify the performance off the proposed method, mAP@0.5, mAP@0.5:0.95, inference time, and the numbers of parameters and FLOPs were used as measurement indicators. Precision was defined as the ratio of positive samples to the total number of samples predicted as positive. Recall was calculated as the proportion of correctly predicted targets to the total number of actual targets.

According to precision and recall, the average precision (AP) and the mean average precision (mAP) could be calculated as follows:

$$AP = \int_0^1 P(R)dR \tag{16}$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \tag{17}$$

where mAP@0.5 refers to the mAP calculated at an IOU threshold of 0.5 for all categories, while mAP@0.5:0.95 refers to the average mAP across different IOU thresholds ranging from 0.5 to 0.95 with a step size of 0.05. The inference time was the time of forward propagation.

4.2. Experimental Results and Comparisons

We compared our proposed method with the YOLOv5l algorithm in numbers of parameters and FLOPs, mAP@0.5, mAP@0.5:0.95, and inference time. As shown in Tables 1 and 2, the results of the ablation experiment are as follows.

Table 1 shows the effect of the different methods on model performance in numbers of parameters and FLOPs and inference time. The numbers of parameters and FLOPs when the backbone network of the baseline was replaced were 21.8 M and 40.2 G, respectively, which were 52.8% and 62.7% less than those for YOLOv5l, and the inference speed was 35.7 ms lower than for YOLOv5l. The ME-YOLOv5 method, based on M-YOLOv5 but replacing the SE module with the ECA attention mechanism, achieved faster inference speed than M-YOLOv5. By redesigning the parameters of the neck structure and adding a prediction head, we obtained the MEL-YOLOv5-S and MEL-YOLOv5-L models. Compared with ME-YOLOv5, the numbers of parameters and FLOPs were reduced by 16.9 M and 30.6 G for MEL-YOLOv5-S, and the inference speed was faster than for ME-YOLOv5. There was a slight increase in the number of FLOPs for MEL-YOLOv5-L compared to ME-YOLOv5, but there were fewer parameters than for ME-YOLOv5. On the other hand, ME-YOLOv5 had a faster inference speed than YOLOv5l.

Table 1. Performance comparison of different methods in numbers of parameters and FLOPs and inference time.

V3	ECA	SL	Method	Params (M)	FLOPs (G)	Inference Time (ms)
			YOLOv5l (baseline)	46.2	107.9	65.5
✓			M-YOLOv5	21.8	40.2	35.7
✓	✓		ME-YOLOv5	20.3	40.2	34.5
✓	✓	✓	MEL-YOLOv5-S	**3.4**	**9.6**	**21.8**
✓	✓	✓	MEL-YOLOv5-L	5.8	56.8	43.5

Bold represents the optimal value of the current column. V3 denotes the MobileNetV3 backbone network, ECA denotes the efficient channel attention mechanism, and SL denotes small object layer.

Table 2. Performance comparison between CIoU and FocalEIoU.

Method	mAP@0.5(%)	mAP@0.5:0.95(%)
MEL-YOLOv5-S	34.8	18.3
MELF-YOLOv5-S	**34.8**	**18.7**
MEL-YOLOv5-L	46.8	26.8
MELF-YOLOv5-L	**46.9**	**27.7**

Bold represents the optimal value of the current column.

From the experimental results in Table 2, the mAP@0.5 and mAP@0.5:0.95 of the MELF-YOLOv5-S method based on the FocalEIoU loss function reached 34.8% and 18.7%, respectively, which were better than the values based on CIoU. In addition, compared with MEL-YOLOv5-L based on CIoU, the mAP@0.5:0.95 of MELF-YOLOv5-L based on FocalEIoU was 0.9% higher.

4.3. Performance Comparison of Mainstream YOLO Series Algorithms

To further validate the performance of the proposed method, we compared it with other YOLO series detection algorithms using different input sizes on the validation set. The results are shown in Table 3 and Figure 9.

Table 3 reveals that the MELF-YOLOv5-S method achieved a state-of-the-art balance between the numbers of parameters and FLOPs when compared to YOLOv3-Tiny, YOLOv4-Tiny, and YOLOv5s. MELF-YOLOv5-S was not as fast as YOLOv4-Tiny in inference speed, but the mAP@0.5 and mAP@0.5:0.95 were significantly superior to those of other lightweight algorithms. In addition, the numbers of parameters and FLOPs of the MELF-YOLOv5-L method were significantly less than those of other large model algorithms, and the inference time of MELF-YOLOv5-L was lower than that of other algorithms.

Figure 9 shows that, compared with the lightweight YOLO series algorithms, MELF-YOLOv5-S had the highest accuracy in mAP@0.5, and MELF-YOLOv5-L also had the highest accuracy compared to other YOLO series methods. In addition, compared with the same scale model, the proposed method had a faster inference speed than other YOLO series algorithms.

Table 3. Performance comparison of YOLO series algorithms with different input sizes.

Method	Size	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)	FLOPs (G)	Inference Time (ms)
	448	11.5	4.8			19.4
YOLOv3-Tiny	640	16.2	7.0	8.7	12.9	21.2
	832	19.6	8.4			23.0
	446	16.6	9.0			**15.5**
YOLOv4-Tiny	640	24.4	13.5	5.9	16.2	**17.7**
	832	29.8	16.7			**22.7**
	446	26.5	13.8			20.0
YOLOv5s	640	33.5	17.8	7.0	15.9	21.5
	832	37.1	20.0			24.6
	446	**29.1**	**14.4**			20.4
MELF-YOLOv5-S	640	**34.8**	**18.7**	**3.4**	**9.8**	21.8
	832	**39.3**	**21.2**			29.6
	448	32.8	17.7			53.4
YOLOv3	640	40.3	22.3	61.5	154.9	81.5
	832	43.4	24.3			121.2
	446	36.5	21.3			67.2
YOLOv4	640	45.3	27.3	64.0	141.6	67.6
	832	**50.4**	**30.8**			83.3
	446	31.5	17.3			44.2
YOLOv5l	640	39.6	22.5	46.2	107.9	65.5
	832	43.4	25.0			99.1
	446	33.2	18.6			78.5
YOLOv5x	640	41.0	23.6	86.2	204.2	118.2
	832	44.9	26.0			175.5
	446	**39.0**	**22.4**			**33.7**
MELF-YOLOv5-L	640	**46.9**	**27.7**	**5.8**	**56.8**	**43.5**
	832	50.2	30.2			**63.9**

Bold represents the optimal value of the current column.

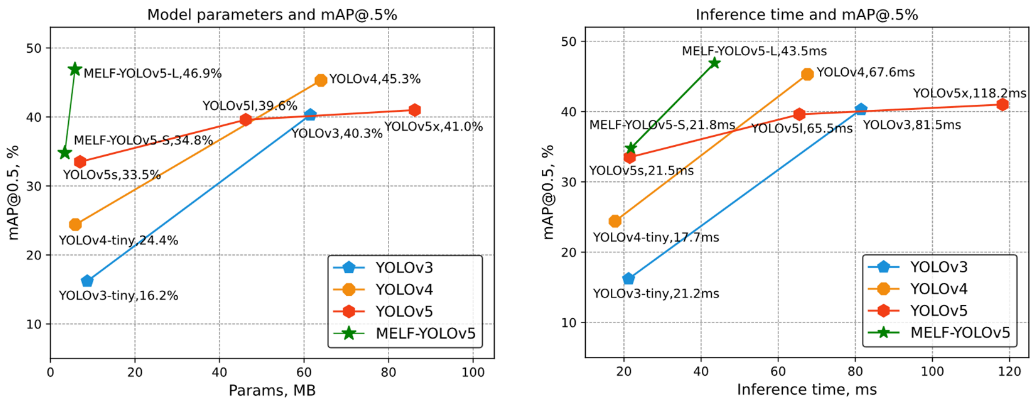


Figure 9. The performance comparison of YOLO series algorithms. (**a**) Shows the relationship between the number of parameters and mAP@0.5; (**b**) represents the relationship between the inference time and mAP@0.5.

Figure 10 shows the performance comparison of MELF-YOLOv5-S and lightweight YOLO series algorithms in different scenes. In the original pictures, YOLOv3-Tiny, YOLOv4-Tiny, and YOLOv5s missed the detection of a large number of objects shown in the red boxes. In the first row of Figure 10, our proposed MELF-YOLOv5-S method showed significant superiority in pedestrian detection compared to other compared algorithms. In the red box of the second row, our proposed method had higher detection performance for tricycles and motorcycles than other YOLO series algorithms. In the last row of Figure 10, our method significantly outperformed other mainstream algorithms in detecting small distant objects, according to the comparison results.

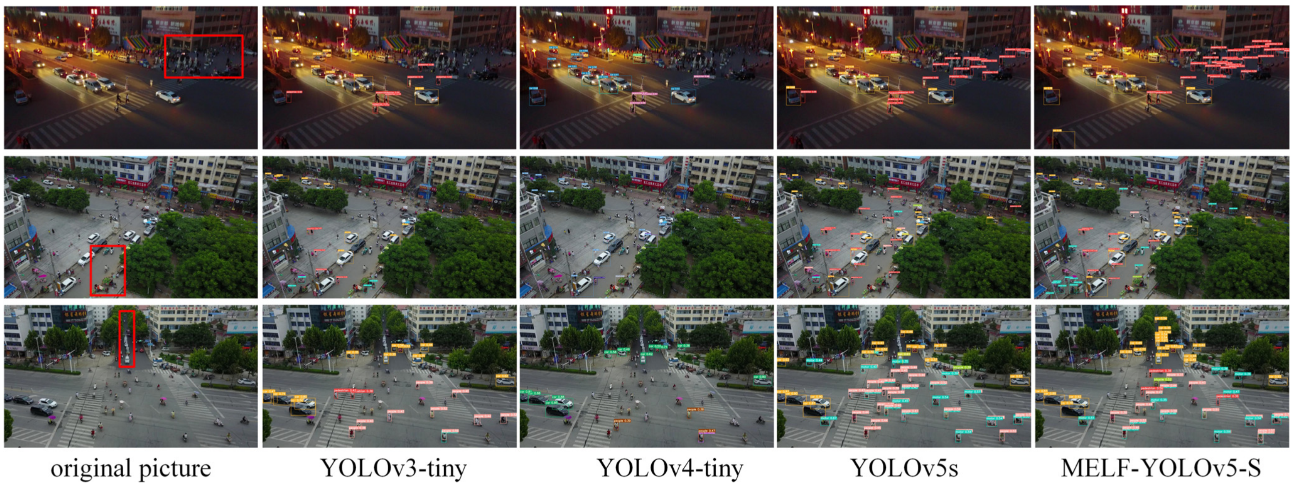


Figure 10. Visual detection results of MELF-YOLOv5-S and other lightweight YOLO series algorithms.

For YOLOv3, YOLOv4, and YOLOv5l, compared to their lightweight versions, the detection performance was significantly improved for pedestrians from the perspective of UAVs, as shown in Figure 11. However, there were still many missed objects in the red boxed areas. Our proposed method further improved the detection ability for small objects. In the red boxes of the second and third rows of Figure 11, although our proposed method showed a slight improvement in detection performance compared to other mainstream YOLO series algorithms, our method had fewer parameters, lower computation complexity, and faster inference speed.

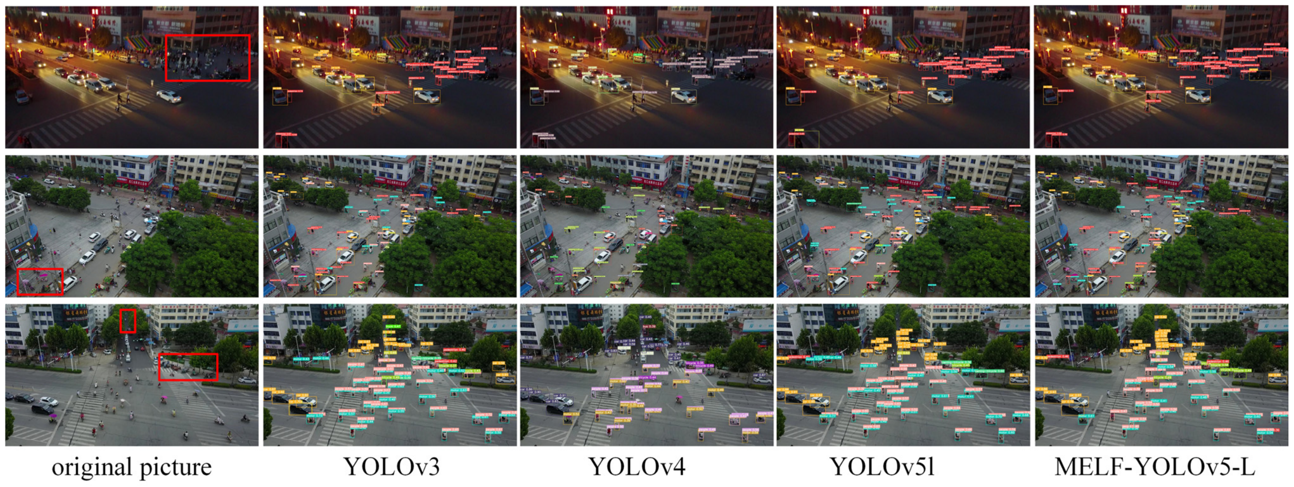


Figure 11. Visual detection results of MELF-YOLOv5-L and other large model YOLO series algorithms.

5. Conclusions

A lightweight, real-time detection algorithm based on YOLOv5l was proposed in this paper, which not only guaranteed high detection accuracy but also improved the inference speed. The proposed method obviously had a reduced number parameters and FLOPs. To reduce the model's overall numbers of parameters and FLOPs, MobileNetV3 based on the ECA attention mechanism was introduced into the backbone network of YOLOv5l. The improved M-YOLOv5 decreased the numbers of parameters and FLOPs by 25.9 M and 67.7 G, respectively, and the inference speed was 47.3% faster than YOLOv5l. To meet the needs of different UAV image scenarios and embedded devices, two kinds of neck structures were designed. MEL-YOLOv5-S was a method with fewer parameters and FLOPs and a small object detection layer was added. It can be seen from Table 1 that MEL-YOLOv5-S only had 4.8 M parameters and 9.8 G FLOPs. Although MEL-YOLOv5-L had 56.8 G FLOPs, the number of parameters was 5.8 M, which was 87.4% lower than YOLOv5l. To address the issue of incorrectly enlarging the side length between the prediction and ground truth boxes, FocalEIoU was introduced into MELF-YOLOv5 based on YOLOv5l. As shown in Table 2, the FocalEIoU loss function was slightly superior to CIoU. Finally, we compared the proposed methods with mainstream YOLO series algorithms using different input sizes. Compared to other lightweight YOLO series algorithms, MELF-YOLOv5-S reached the highest mAP@0.5 and mAP@0.5:0.95 and achieved real-time inference speed. MELF-YOLOv5-L had a faster inference speed than other large model YOLO series algorithms. To sum up, this paper provides a lightweight, real-time algorithm to balance inference speed and detection accuracy. Future work will primarily focus on two aspects: further compressing the network parameters and FLOPs and enhancing the feature extraction ability of the lightweight mode through re-parameterization techniques.

Author Contributions: Conceptualization, P.S. and Y.W.; Data curation, P.S.; Formal analysis, P.S.; Funding acquisition, L.C.; Methodology, P.S.; Resources, L.C.; Supervision, L.C.; Validation, P.S.; Visualization, P.S. and B.P.; Writing—original draft, P.S. and Y.Y.; Writing—review & editing, P.S. and Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Industry-University-Research Innovation Fund of China University (2021ZYA11002), the Sichuan Science and Technology Program (No. 2020YJ0368), the Opening Project of Sichuan Province University Key Laboratory of Bridge Non-destruction Detecting and Engineering Computing (No. 2021QZJ01), and the Graduate Innovation Fund Project of Sichuan University of Science & Engineering (No. Y2022107).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Viola, P.; Jones, M. Rapid Object Detection Using a Boosted Cascade of Simple Features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001; p. I-I. [CrossRef]
2. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
3. Felzenszwalb, P.; McAllester, D.; Ramanan, D. A discriminatively trained, multiscale, deformable part model. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
4. Wang, Z.; Qi, L.; Tie, Y.; Ding, Y.; Bai, Y. Drone Detection Based on FD-HOG Descriptor. In Proceedings of the 2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Zhengzhou, China, 18–20 October 2018; pp. 433–4333.
5. Xu, Y.; Yu, G.; Wang, Y.; Wu, X.; Ma, Y. A Hybrid Vehicle Detection Method Based on Viola-Jones and HOG + SVM from UAV Images. *Sensors* **2016**, *16*, 1325. [CrossRef] [PubMed]
6. Jiang, J.; Zhong, X.; Chang, Z.; Gao, X. Object Detection of Transmission Tower Based on DPM. In Proceedings of the 4th International Conference on Information Technologies and Electrical Engineering, Changde, China, 4–6 November 2023; p. 24.
7. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. Acm* **2012**, *60*, 84–90. [CrossRef]
8. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409-1556.
9. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.E.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
10. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
11. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.

12. Uijlings, J.R.R.; van de Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective Search for Object Recognition. *Int. J. Comput. Vision* **2013**, *104*, 154–171. [CrossRef]
13. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
15. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. In You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
16. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
17. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804-2767.
18. Bochkovskiy, A.; Wang, C.; Liao, H.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004-10934.
19. Wang, C.; Liao, H.M.; Wu, Y.; Chen, P.; Hsieh, J.; Yeh, I. In CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
20. Wang, Z.; Zhang, X.; Li, J.; Luan, K. A YOLO-Based Target Detection Model for Offshore Unmanned Aerial Vehicle Data. *Sustainability* **2021**, *13*, 12980. [CrossRef]
21. Li, S.; Li, Y.; Li, Y.; Li, M.; Xu, X. YOLO-FIRI: Improved YOLOv5 for Infrared Image Object Detection. *IEEE Access* **2021**, *9*, 141861–141875. [CrossRef]
22. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. In TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-Captured Scenarios. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision Workshops, Montreal, Canada, 10–17 October 2021; pp. 2778–2788.
23. Li, Y.; Yuan, H.; Wang, Y.; Xiao, C. GGT-YOLO: A Novel Object Detection Algorithm for Drone-Based Maritime Cruising. *Drones* **2022**, *6*, 335. [CrossRef]
24. Li, W.; Wu, G.; Sun, H.; Bai, C.; Bao, W. In Dim and Small Target Detection in Unmanned Aerial Vehicle Images. In Proceedings of the 2022 International Conference on Autonomous Unmanned Systems, Singapore, 23–25 September 2023; pp. 3143–3152.
25. Cheng, Q.; Wang, H.; Zhu, B.; Shi, Y.; Xie, B. A Real-Time UAV Target Detection Algorithm Based on Edge Computing. *Drones* **2023**, *7*, 95. [CrossRef]
26. Lin, T.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
27. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
28. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef] [PubMed]
29. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. In Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the 2020 AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12993–13000.
30. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704-4861.
31. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
32. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.; Tan, M.; Chu, G.; Vasudevan, V.; Brain, G.; Zhu, Y.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
33. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef] [PubMed]
34. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Hu, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11531–11539.
35. Zhang, Y.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* **2022**, *506*, 146–157. [CrossRef]
36. Cao, Y.; He, Z.; Wang, L.; Wang, W.; Yuan, Y.; Zhang, D.; Zhang, J.; Zhu, P.; Gool, L.V.; Han, J.; et al. VisDrone-DET2021: The Vision Meets Drone Object detection Challenge Results. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision Workshops, Montreal, BC, Canada, 10–17 October 2021; pp. 2847–2854.