Chinese Journal of Electronics
vol. 33, no. 4, pp. 997–1009, July 2024
https://doi.org/10.23919/cje.2022.00.300

RESEARCH ARTICLE

Lightweight Object Detection Networks for UAV Aerial Images Based on YOLO

Yanshan LI[1,2], Jiarong WANG[1,2], Kunhua ZHANG[1,2], Jiawei YI[1,2], Miaomiao WEI[1,2], Lirong ZHENG[1,2], and Weixin XIE[1,2]

1. *ATR National Key Laboratory of Defense Technology, Shenzhen University, Shenzhen 518000, China*
2. *Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518000, China*

Corresponding author: Yanshan LI, Email: lys@szu.edu.cn

Manuscript Received September 2, 2022; Accepted August 7, 2023

Abstract — Existing high-precision object detection algorithms for UAV (unmanned aerial vehicle) aerial images often have a large number of parameters and heavy weight, which makes it difficult to be applied to mobile devices. We propose three YOLO-based lightweight object detection networks for UAVs, named YOLO-L, YOLO-S, and YOLO-M, respectively. In YOLO-L, we adopt a deconvolution approach to explore suitable upsampling rules during training to improve the detection accuracy. The convolution-batch normalization-SiLU activation function (CBS) structure is replaced with Ghost CBS to reduce the number of parameters and weight, meanwhile Maxpool maximum pooling operation is proposed to replace the CBS structure to avoid generating parameters and weight. YOLO-S greatly reduces the weight of the network by directly introducing CSPGhostNeck residual structures, so that the parameters and weight are respectively decreased by about 15% at the expense of 2.4% mAP. And YOLO-M adopts the CSPGhostNeck residual structure and deconvolution to reduce parameters by 5.6% and weight by 5.7%, while mAP only by 1.8%. The results show that the three lightweight detection networks proposed in this paper have good performance in UAV aerial image object detection task.

Keywords — Aerial images, Object detection, Deep learning, You only look once, Lightweight network.

Citation — Yanshan LI, Jiarong WANG, Kunhua ZHANG, *et al.*, “Lightweight Object Detection Networks for UAV Aerial Images Based on YOLO,” *Chinese Journal of Electronics*, vol. 33, no. 4, pp. 997–1009, 2024. doi: 10.23919/cje.2022.00.300.

I. Introduction

In recent years, with the rapid development of UAV technology, the study of UAV image has become a hot research hotspot [1]–[5]. However, UAV aerial images are highly unstructured and cover wide areas, making it difficult to separate the target and background. Besides, UAV needs miniaturization of load and low energy consumption. It can be seen that UAV aerial image object detection based on deep learning faces many challenges. Therefore, it is important to study the object detection algorithm of UAV aerial image [6]–[10]. For small target detection or overlapping target detection tasks, some researches introduce the multi-mode fusion method [11]–[13], which uses the information of multiple modes to improve the detection performance. Unfortunately, UAV aerial image target detection networks with high detection accuracy are often characterized by many parameters and heavy weight, which poses certain obstacles to the application of deep learning-based target detection networks on UAV aerial images.

To solve this problem, researchers have proposed a series of lightweight deep networks, including MobileNet [14]–[16], ShuffleNet [17], [18], GhostNet [19], [20], YOLOv5n [21], and so on. Among them, networks such as YOLOv5n are lightweight networks oriented to target detection, and the results on natural scene image datasets show that it sharply reduces the number of network parameters and ensures a certain detection accuracy compared with YOLO networks.

According to the characteristics of UAV aerial images, we propose three lightweight networks for UAV aerial image target detection under the framework of YOLOv5n, named YOLO-L, YOLO-M, and YOLO-S.

Associate Editor: Prof. Meng WANG, Hefei University of Technology.

The main contributions of this paper are as follows:

1) A lightweight CSPGhostNeck residual structure is designed. Using the CSPGhostNeck residual structure to fuse the sparse features of aerial images can effectively reduce the number of parameters and weight of the detection network.

2) Combining the CSPGhostNeck residual structure and deconvolution, three lightweight networks with different number of parameters, YOLO-L, YOLO-M, and YOLO-S, are proposed, respectively. Compared to the baseline network, the three proposed networks all exhibit the capability to reduce network weight to varying degrees. However, different network architectures have varying impacts on accuracy. This allows for the application of these three networks in different scenarios where a balance between accuracy and weight reduction is required.

Compared to the baseline network, YOLO-L introduces deconvolution to enhance the accuracy, and replaces the convolution-batch normalization-SiLU activation function (CBS) structure with Ghost CBS to reduce the number of parameters and weight. At the same time, the Maximum pooling operation is used to avoid the increase of parameter and weight. Therefore, the accuracy is improved while the weight of the network is reduced, which allows YOLO-L to be applied to the situation that accuracy needs to be improved while weight and parameters reduced. And the proposed YOLO-S introduces the CSPGhostNeck residual structure to prevent gradient from disappearing, which enhances the feature extraction ability of the network, and the parameters and weight can be greatly reduced compared with the baseline network. Therefore, YOLO-S can meet the needs of applications aiming at a significant reduction in parameters and weight, but can tolerate a slight reduction in accuracy. The YOLO-M adopts the combination of the CSPGhostNeck residual structure and deconvolution and obtains higher detection accuracy while reducing the number of parameters and weight of the network. Compared with YOLO-S, YOLO-M has less accuracy loss, but at the same time the reduction of parameters and weight is relatively weak. Therefore, YOLO-M is suitable for applications that require reduced weight and parameters, but need the smallest possible accuracy loss.

II. YOLO-M

This paper firstly elaborates on the YOLO-M network for UAV aerial images and then introduces the other networks.

1. Overall network architecture

The YOLO-M mainly consists of feature extraction, feature fusion [22], prediction, and post-processing stages. Figure 1 shows the overview of our method, where CBS stands for convolution-batch normalization-SiLU activation function. We can set the convolution parameters of CBS to adjust the channel of the input image or downsample the input image. SPPF [20] is a spatial pyramid pooling structure, which consists of CBS structure and Maxpool maximum pooling operation. The output of SPPF keeps the same size and channel as the input. Besides, Head represents prediction head, CG represents the proposed CSPGhostNeck residual structure, and CB represents CSPBottleNeck residual structure. Red CBS represents setting the convolution parameters of CBS to downsample. Gray CBS indicates setting the convolution parameters of CBS to adjust the channel of the input image.

1) In the feature extraction stage, we use the CSPBottleNeck residual structure, the CBS structure, and the SPPF structure to capture feature effectively in low-resolution aerial images. To detect targets of different scales, the feature extraction network downsamples the input image five times to obtain feature maps of differ-

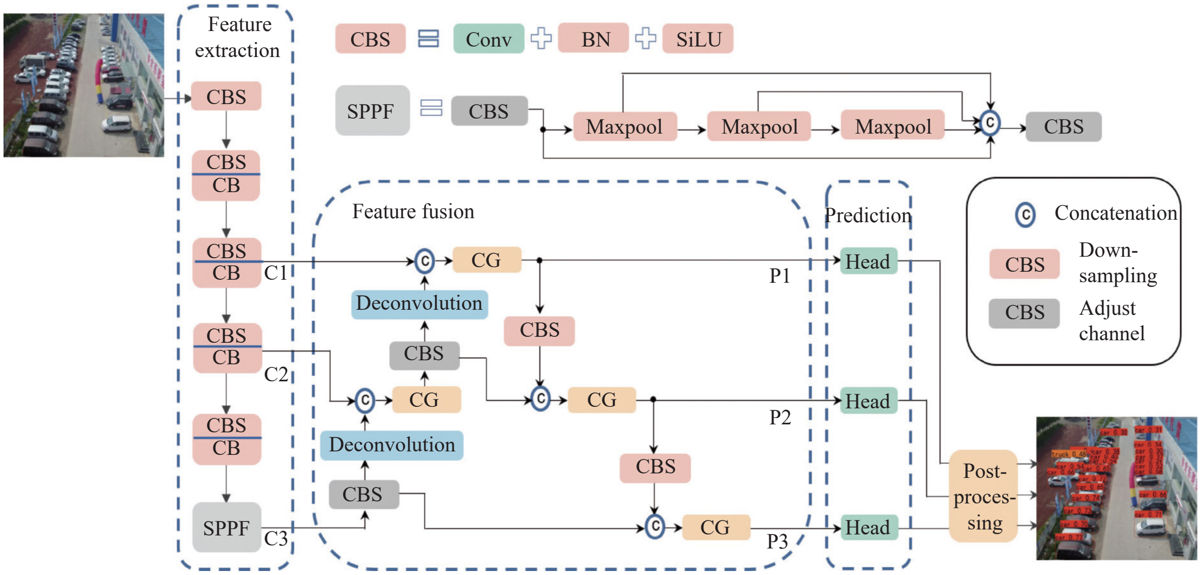


Figure 1 Network architecture of YOLO-M.

ent sizes and retains the last three of them. The downsampling is implemented by CBS structure, setting the convolutional kernel size to 3×3, stride to 2, and padding to 1. After 5 times of downsampling, the width and height of the minimum feature map become 1/32 of the width and height of the input image. Since downsampling decreases the image size and loses the effective features, we supplement the features with the dimension of the feature map. As downsampling proceeds, the dimensionality of the feature map gradually increases. The CBS structure that plays the role of downsampling uses the SiLU activation function. Its expression is shown as

$$\mathrm{SiLU}(x) = x \cdot \sigma(x) \tag{1}$$

where x is the input image and $\sigma(\cdot)$ denotes the sigmoid function, i.e., $\sigma(\cdot) = 1/(1 + \mathrm{e}^{-x})$

The residual structure after the downsampling operation can deepen the network and avoids gradient disappearance. The residual structure has different representations. YOLO-M uses the CSPBottleNeck residual structure for efficient extraction of deep features. After the residual structure, the smallest feature map is fed to the SPPF structure, containing three 5×5 maxpooling operations. The SPPF can increase the perceptual field and give the network a comprehensive perception of the image. The results of the last three downsampling and the output of SPPF are taken as the input of the feature fusion stage.

2) In the feature fusion stage, the PANet network is used to fuse the extracted features, whose component units include CBS structure, CSPGhostNeck residual structure, and deconvolution and splicing operations. The functions of the CBS structure include adjusting the dimensionality of feature map and downsampling. The PANet network contains two channels, the bottom-up channel and the top-down channel. The bottom-up channel passes the information from the lower layer to the upper layer, and the top-down channel does the same thing in the opposite direction. Since the size of feature maps varies from layer to layer, the feature maps need to be upsampled or downsampled before the feature maps are transferred. In YOLO-M, the deconvolution is introduced and the up-sampling operation is accomplished by setting the parameters of the deconvolution. Since the deconvolution has learnable parameters, the detection network can learn upsampling rules effectively during the training. The learnable parameters enable the network to reduce information loss, conduct feature fusion better, and improve detection accuracy. The downsampling in YOLO-M is done through the CBS structure.

After connecting the upper and lower feature maps, they are sent to the residual structure for fusion to reduce the confounding effect. To reduce the number of parameters and weight of the network, a lightweight CSPGhostNeck residual structure is designed. YOLO-M uses the CSPGhostNeck residual structure to fuse the connected features, thus effectively reducing the size of the network.

3) In the prediction stage, three prediction heads are used to predict targets of different scales for the characteristic of large scale span of UAV aerial image targets. The prediction head contains the operations of convolution and rearrangement matrix, which can adjust the dimension of the feature map to a uniform value. Let the number of categories in the dataset be C, the offset value be 4, the number of the prior box be n, and the calculation process of the adjusted dimension is shown in (2).

$$C = n \times (c + k + 4) \tag{2}$$

where k represents whether the prior box contains the target or not, and c is the number of categories, which represents the probability of the prior box belongs to each category.

4) In the post-processing stage, the 3 prediction heads are post-processed to filter out the valid priori boxes. The confidence threshold is set to remove the priori boxes with confidence less than the threshold. According to the NMS algorithm, we set the IoU threshold to compare the intersection and merging ratio of the priori frames with the ground true. For multiple prior boxes with intersection ratio higher than the IoU threshold, the one with the highest value is retained. The post-processing outputs of the three prediction heads are aggregated to get the results.

2. CSPGhostNeck residual structure

In order to construct a lightweight network, this paper designs a residual structure named CSPGhostNeck.

1) CSPGhostNeck residual structure

The structure of CSPGhostNeck is shown in Figure 2, where CBS means convolution-batch normalization-SiLU activation operation, DWCBS indicates deep convolution-batch normalization-SiLU activation operation, Conv represents convolution operation, Flatten means spreading operation, Concat represents splicing operation, Add indicates summation operation, and Multiply denotes multiplication operation.

The overall structure of CSPGhostNeck is shown in Figure 2(a), which contains two branches. One branch performs 1×1 convolution to reduce the dimension of the input feature by half, and the other first feeds the input feature into the Ghost CBS structure to reduce its dimension, and then feeds the output into the GhostNeck structure. The results of the two branches are concatenated together, and then fed into the 1×1 CBS structure to restore the dimension and get the output feature. The width, height, and dimension of the final output feature do not change, and the function of CSPGhostNeck is to fuse the concatenated features and reduce the confounding effect. Using the CSPGhostNeck residual structure can effectively reduce the number of parameters and weight of the detection network. The calculation of CSPGhostNeck is shown in (3), where x is

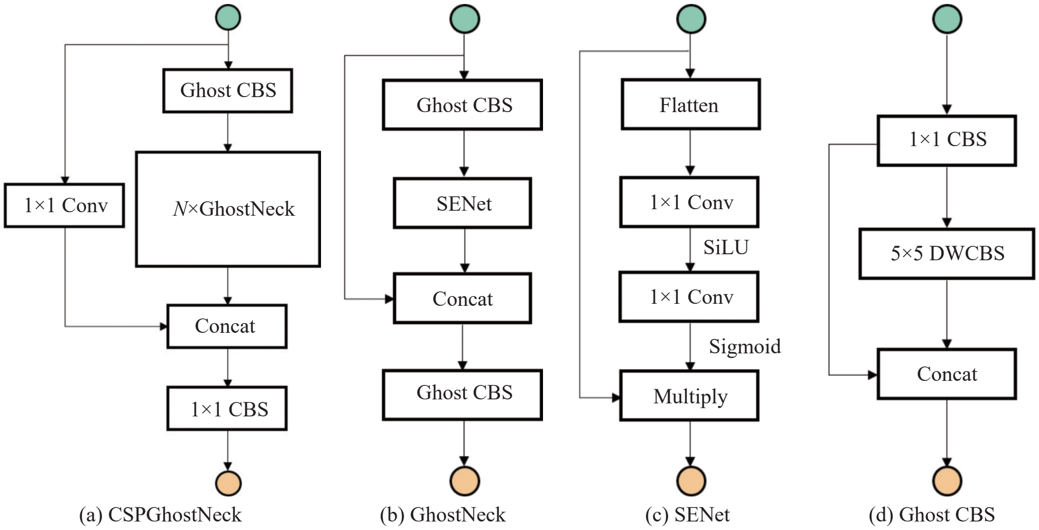


Figure 2 Structure of CSPGhostNeck.

the input feature and y is the output feature.

$$y = \mathrm{CBS}_{1\times1}\Big(\mathrm{Concat}\big(\mathrm{Conv}_{1\times1}(x), \mathrm{GhostNeck}^{N}(\mathrm{GhostCBS}(x))\big)\Big) \tag{3}$$

Figure 2(b) depicts the GhostNeck structure, which plays an important role in CSPGhostNeck. GhostNeck sends the input feature into the 1×1 Ghost CBS structure to reduce the dimension, and sends the result into the SENet structure for dimension weighting. Finally, the output of SENet and the input feature are concatenated and sent into the Ghost CBS structure to recover the dimension and get the final output feature. The calculation of GhostNeck is shown in (4).

$$y = \mathrm{GhostCBS}\Big(\mathrm{Concat}\big(x, \mathrm{SENet}(\mathrm{GhostCBS}(x))\big)\Big) \tag{4}$$

Figure 2(c) shows the SENet structure in GhostNeck, which is an attention mechanism. SENet flattens the input features and feeds them into a 1×1 convolution with a SiLU activation function to reduce the dimensionality. Then it feeds the output into a 1×1 convolution with a sigmoid activation function to recover the dimension and get the dimensional weights. Finally, the dimension-weighted output feature is obtained by multiplying the input feature with the dimensional weights. The width, height, and dimension of the features remain unchanged after being processed by the SENet structure. The calculation process of SENet is shown in (5).

$$y = \mathrm{Multiply}\Big(x, \mathrm{Conv}_{1\mathrm{x}1,\mathrm{Sigmoid}}\big(\mathrm{Conv}_{1\mathrm{x}1,\mathrm{SilU}}(\mathrm{Flatten}(x))\big)\Big) \tag{5}$$

Figure 2(d) shows the Ghost CBS structure, which is involved in the construction of both CSPGhostNeck and GhostNeck. Firstly, Ghost CBS feeds the input feature into a 1×1 CBS structure, which changes the dimension to half of the output dimension. Then it is divided into two branches, where one is left unprocessed and the other feeds the output of the previous step into a 5×5 DWCBS structure for deep convolution. The results of the two branches are concatenated together to obtain the output feature. After being processed by the Ghost CBS structure, the width and height of the input and output features are kept the same, and the dimension of the output feature is set as needed. The calculation process of Ghost CBS is shown in (6).

$$y = \mathrm{Concat}\big(\mathrm{CBS}_{1\times1}(x), \mathrm{DWCBS}_{5\times5}(\mathrm{CBS}_{1\times1}(x))\big) \tag{6}$$

2) Theoretical analysis of the CSPGhostNeck lightweighting

CSPGhostNeck can be regarded as the combination of GhostNeck and CSPNet. GhostNeck is the internal module of CSPGhostNeck, while CSPNet is the external framework of CSPGhostNeck. And BottleNeck can achieve the function of GhostNeck, while CommonNet has similiar effects to CSPNet. The differences are reflected in the structure, number of parameters, and weight.

The network parameters are stored in the storage device in the form of bytes, which is visualized as a weight file. Therefore, the number of parameters is positively correlated with the weight. We compare the structure and the number of parameters respectively between GhostNeck and BottleNeck as well as CSPNet and CommonNet. It is proved at the theoretical level that GhostNeck and CSPNet can effectively reduce the number of parameters and weight of the network. So CSPGhostNeck is a lightweight residual structure.

3) Comparison of GhostNeck and BottleNeck

The GhostNeck and BottleNeck designed in this paper belong to the same residual structure, which can deepen the network and avoid gradient disappearance. The structures of GhostNeck and BottleNeck are shown in Figure 3.

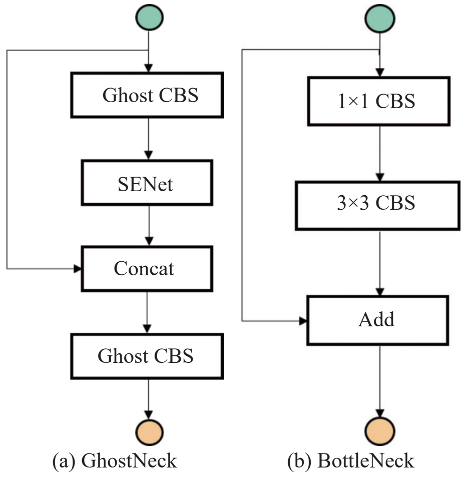


Figure 3 Structure of GhostNeck and BottleNeck.

Firstly, by comparing these two structures, we find that GhostNeck reduces dimension by Ghost CBS, while BottleNeck reduces dimension by 1×1 CBS. GhostNeck uses the attention mechanism, while BottleNeck does not. GhostNeck stitches the input feature with the intermediate result, and then sends it into Ghost CBS to restore dimension, while BottleNeck restores dimension by 3×3 CBS and finally adds the input feature with the intermediate results. The size and dimension of the feature do not change both in GhostNeck and BottleNeck.

Neither concatenation nor summation operations generate additional parameters, and the network parameters are generated primarily by the convolution operation. Set the dimension of the input as C_{in}, the dimension of the output as C_{out}, and the size of the convolution kernel as $n \times n$ with no bias. The number of parameters of a CBS is shown as follows:

$$\text{Params} = C_{\text{in}} \times n \times n \times C_{\text{out}} \tag{7}$$

The number of parameters of a DWCBS is presented as follows:

$$\text{Params} = C_{\text{in}} \times n \times n \tag{8}$$

For a SENet with a fixed convolution kernel size of 1×1 and an output dimension equal to the input dimension, its number of parameters is as follows:

$$\text{Params} = 2 \times C_{\text{in}} \times C_{\text{med}} \tag{9}$$

where C_{med} is the intermediate dimension, and C_{in} is equal to the input dimension multiplied by the compression rate ratio, i.e. $C_{\text{med}} = C_{\text{in}} \times \text{ratio}$.

For GhostNeck and BottleNeck, the input and output dimensions are both. The 1×1 CBC structure halves the feature dimension and the 5×5 DWCBS structure keeps the dimension unchanged, while the 3×3 CBS structure doubles the dimension. In the SENet structure, the compression rate is set to 0.5, which represents that the dimension of the intermediate feature map is equal to half of the dimension of the input feature. Then the number of parameters of GhostNeck is shown in (10).

$$\begin{aligned} P_{\text{Ghost}} =& \left(C \times 1 \times 1 \times \frac{C}{4} + \frac{C}{4} \times 5 \times 5\right) + \left(2 \times \frac{C}{2} \times \frac{C}{4}\right) \\ &+ \left(\frac{3C}{2} \times 1 \times 1 \times \frac{C}{2} + \frac{C}{2} \times 5 \times 5\right) \\ =& \frac{5C^2}{4} + \frac{75C}{4} \end{aligned} \tag{10}$$

Equation (11) depicts the number of parameters of BottleNeck.

$$P_{\text{Bottle}} = C \times 1 \times 1 \times \frac{C}{2} + \frac{C}{2} \times 3 \times 3 \times C = 5C^2 \tag{11}$$

The ratio of the number of parameters of GhostNeck and BottleNeck is as follows:

$$R = \frac{P_{\text{Ghost}}}{P_{\text{Bottle}}} = \frac{\frac{5C^2}{4} + \frac{75C}{4}}{5C^2} = \frac{15}{4C} + \frac{1}{4} \tag{12}$$

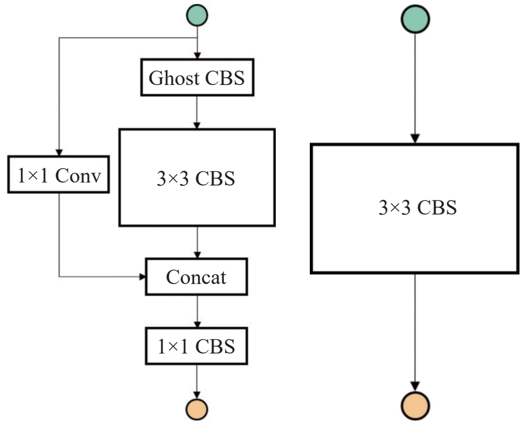
In the actual network, the value of C is $\{64, 128, 256\}$, then the value of R is $\{0.26, 0.28, 0.31\}$, and the minimum value of R is 0.26, while the maximum value of it is 0.31. Using GhostNeck instead of BottleNeck, the number of parameters is at least 0.26 times of the original, and the maximum number becomes the original 0.31 times. This shows GhostNeck can significantly reduce the number of parameters, so the network becomes lighter.

4) Comparison of CSPNet and CommonNet

CSPNet is a two-branch structure, where the primary branch can feed the input feature into any network unit and the secondary branch is adjusted as needed; CommonNet is a single-branch structure, which feeds the input feature directly into the network unit. The idea of CSPNet can be implemented in different forms. For the CSPNet implemented in this section, the component units include Ghost CBS structure, CBS structure, Conv operation, concatenate operation, and the main branch unit. The similarities and differences between CSPNet and CommonNet are shown in Figure 4 by comparing two network structures, with 3×3 CBS as the main branch unit as an example.

Firstly, the structures of the two are compared. CSPNet contains two branches. The main branch of it feeds the input feature into the Ghost CBS structure to halve its dimension before feeding it into the 3×3 CBS structure, and the secondary branch is convolved with 1×1. The concatenation of the results of the two branches is then fed into a 1×1 CBS structure to recover the dimension and get the output feature. As for CommonNet, it directly feeds the input feature into a 3×3 CBS structure.

Secondly, the number of parameters of both is com-



(a) CSPNet (b) CommonNet

Figure 4 Structures of CSPNet and CommonNet.

pared. Both the input and output dimensions of CSPNet and CommonNet are C. The number of parameters of CSPNet is as follows:

$$\begin{aligned}P_{\text{CSPN}} =& \left(C\times 1\times 1\times \frac{C}{2}\right)+\left(C\times 1\times 1\times \frac{C}{4}\right.\\&\left.+\frac{C}{4}\times 5\times 5\right)+\left(\frac{C}{2}\times 3\times 3\times \frac{C}{2}\right)\\&+(C\times 1\times 1\times C)\\=&4C^2+\frac{25C}{4}\end{aligned}\tag{13}$$

The number of parameters of CommonNet can be expressed as follows:

$$P_{\text{CN}} = C\times 3\times 3\times C = 9C^2\tag{14}$$

The ratio of the number of parameters of CSPNet and CommonNet is as follows.

$$R=\frac{P_{\text{CSPN}}}{P_{\text{CN}}}=\frac{4C^2+\dfrac{25C}{4}}{9C^2}=\frac{25}{36C}+\frac{4}{9}\tag{15}$$

If the dimension of the input feature is $C\geq 3$, the ratio of the number of parameters of CSPNet to CommonNet is $R\leq 0.68$. Compared with CommonNet, CSPNet can effectively reduce the number of parameters and weight, thus effectively reducing the network magnitude.

CSPGhostNeck combines the advantages of GhostNeck and CSPNet, and it is more effective in reducing the number of parameters and weight, making the network more lightweight.

3. Deconvolution

In the feature fusion stage, we need to upsample the high-level feature map in order to fuse the features with the low-level feature map. YOLO-M introduces deconvolution to upsample, so that the high-level feature map can retain more information and fuse with the low-level feature map more effectively.

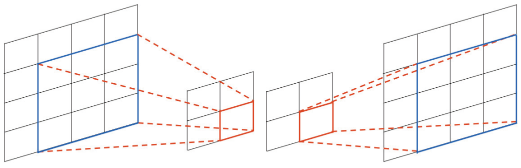
Deconvolution is an inverse procession of convolution. The relationship between the input and output of convolution is many-to-one, while the input and output of deconvolution is a kind of one-to-many relationship. Figure 5 shows the difference between convolution and deconvolution.

Generally, deconvolution is performed on the input feature, and the size of the output feature is as follows:

$$O=(I-1)\times s+k-p\times 2+\text{op}\tag{16}$$

where O is the output size, I is the input size, s is the step size, k is the size of the convolution kernel, p is the input padding, and op is the output padding.

The upsampling operation can be completed by setting parameters for deconvolution. In the lightweight YOLO network of this paper, it is necessary to upsample the high-level feature map twice. Therefore, the size of the convolution kernel of the deconvolution is set to 3×3, the stride is 2, the padding is 1, and the output fill is 1. From (16), it can be seen that the size of the output feature is twice the size of the input feature. Finally, batch normalization and SiLU function activation are performed on the upsampled results to obtain the features of normalized distribution.



(a) Schematic diagram of convolution (b) Schematic diagram of deconvolution

Figure 5 The difference between convolution and deconvolution.

The traditional method uses the nearest-neighbor interpolation for upsampling, and the pixel values of the nearest neighbors are used as the pixel values of the sampling points, which is simple and fast. However, the image is prone to mosaic and jaggedness. The nearest-neighbor interpolation uses a manually set upsampling rule with fixed computational parameters, which is more rigid. Deconvolution allows the detection network to learn the upsampling rule with unfixed computational parameters by itself, providing flexibility and versatility. Compared with the nearest-neighbor interpolation, deconvolution makes the upsampling rules more compatible with the needs of target detection, thus improving detection accuracy.

III. YOLO-L and YOLO-S

In some application scenarios, the network is required to have higher detection accuracy, while in some other application, the network is required to have lower number of parameters and weight. From Section II, it is known that deconvolution can improve the accuracy of detection, but it also increases the number of parameters and weight of the network. CSPGhostNeck residual structure can reduce the complexity of the network, but it weakens the detection performance of the network. On the basis of the characteristics of deconvolution and CSPGhostNeck residual structure, YOLO-L lightweight network with higher detection accuracy and YOLO-S lightweight network with fewer number of parameters and lighter weight are designed.

1. YOLO-L

The structure of YOLO-L is shown in Figure 6, which contains four stages: feature extraction, feature fusion, prediction, and post-processing.

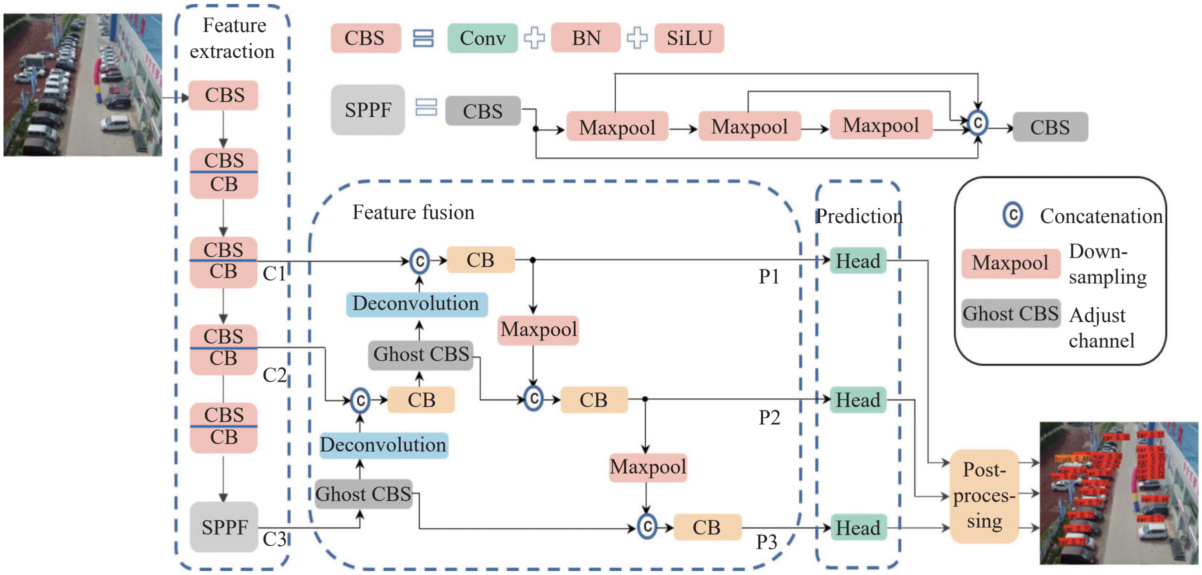


Figure 6 Network architecture of YOLO-L.

Firstly, in the feature extraction stage, YOLO-L extracts features using CBS structure, CSPBottleNeck residual structure, and SPPF structure. The CBS is exploited to downsample and extract features at different levels. The CSPBottleNeck is applied to deepen the network, prevent gradient from disappearance, and enhance the feature extraction ability of the network. The SPPF is used to increase the perceptual field and coalesce the high-level semantic features. After several times of downsampling, five feature maps of different sizes are generated. The smallest feature maps are sent into the SPPF structure to obtain the more condensed semantic information. The results of the last three downsampling and the output of SPPF are retained as the input to the feature fusion network.

Then, in the feature fusion stage, PANet is selected as the feature fusion network, consisting of Ghost CBS structure, CSPBottleNeck residual structure, deconvolution, Maxpool maximum pooling operation, and splicing operation. In the bottom-up pathway of PANet, the high-level feature maps are reduced in dimension by Ghost CBS structure, followed by deconvolution to upsample effectively. The concatenation of the deconvoluted output and the upper feature map is then fed into the CSPBottleNeck residual structure to achieve feature interaction. In the top-down pathway of PANet, the large-size feature map is downsampled using the maximum pooling operation, concatenated with the lower-level feature map, and finally sent to the CSPBottleNeck residual structure to fuse the concatenated features. The outputs of the feature fusion network PANet of three different sizes are as the inputs to the prediction stage.

Finally, in the prediction and post-processing stages, the feature maps of the three sizes are predicted and post-processed, and the outputs of the post-processing are summarized to obtain the detection results.

YOLO-L uses the CSPBottleNeck residual structure, which can fuse features more effectively. Compared with CSPGhostNeck, CSPBottleNeck has a large number of parameters, more weight, more complex computation, and better feature processing. In the feature fusion stage, YOLO-L introduces deconvolution to complete the upsampling operation, which maximizes the information retention of high-level feature maps and enhances the performance of the feature fusion network. Although CSPBottleNeck and deconvolution improve the detection accuracy of the network, it increases the magnitude of the network. To offset the effects of CSPBottleNeck and deconvolution, YOLO-L replaces the CBS structure with Ghost CBS structure for dimension adjustment of the high-level feature map in the feature fusion stage, and reduces the number of parameters and weight. Maxpool maximum pooling operation replaces the CBS structure to achieve the downsampling function without generating additional parameters and weight. Therefore, YOLO-L has high detection accuracy under the condition of lower weight and a smaller number of parameters.

2. YOLO-S

The structure of YOLO-S is shown in Figure 7. Similarly, it contains four stages: feature extraction, feature fusion, prediction, and post-processing. In the figure, Nearest represents the upsampling operation of the nearest-neighbor interpolation.

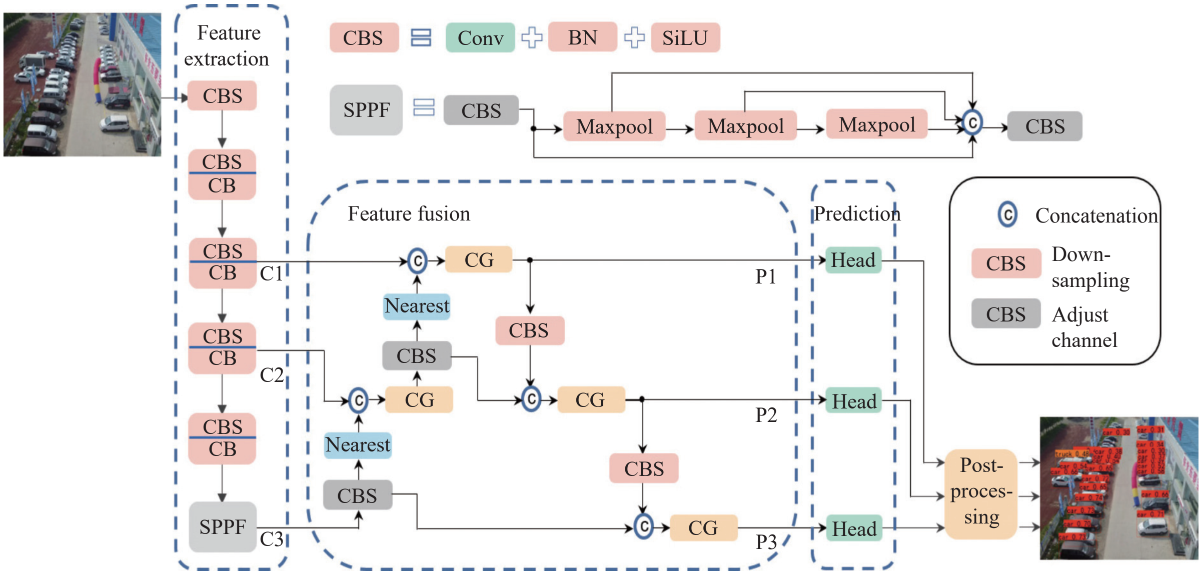


Figure 7 Network architecture of YOLO-S.

In the feature extraction stage, YOLO-S extracts deep-level features using the CBS structure, CSPBottleNeck residual structure, and SPPF structure, and outputs feature maps of three different sizes. In the feature fusion stage, YOLO-S uses PANet as a feature fusion network to fuse the features extracted in the previous stage. In the prediction and post-processing stages, YOLO-S predicts three sizes of feature maps, detects multiple scales of targets through post-processing such as NMS algorithm, and adds target boxes of different scales to obtain detection results.

To achieve fewer parameters and lighter weight, YOLO-S uses the CSPGhostNeck residual structure to fuse features, and adopts the nearest-neighbor interpolation to complete the upsampling operation. Compared with CSPBottleNeck, CSPGhostNeck has fewer parameters, lighter weight, and simpler computation. Compared with deconvolution, the nearest-neighbor interpolation has no learnable parameters and does not increase the number of parameters or weight. Therefore, YOLO-S has fewer parameters, less weight, and a higher level of lightness.

IV. Experiments and Analysis

Firstly, to prove the effectiveness of CSPGhostNeck residual structure and deconvolution, both of them are added to the benchmark network for ablation experiments. Then, to verify the performance of YOLO-L, YOLO-M, and YOLO-S, algorithm comparison experiments are conducted.

1. Dataset

To verify the design of this paper and the effectiveness of YOLO-L, YOLO-M, and YOLO-S, the publicly available datasets VisDrone2021-DET and CARPK are selected for the experiments.

1) Dataset VisDrone2021-DET

VisDrone2021-DET is a UAV aerial feature dataset introduced by the AISKYEYE team of Tianjin University, which is obtained from UAVs equipped with different types of cameras under different scenes, weather, and lighting conditions. The dataset is annotated and contains 10 classes: pedestrians, people, bicycles, cars, vans, trucks, tricycles, tricycles with awnings, buses, and motorcycles. The dataset VisDrone2021-DET is uneven, showing a long-tail distribution. The largest number of samples is cars, with 144625 vehicles, and the smallest number of samples is tricycles with canopies, with only 3244 vehicles. VisDrone2021-DET is divided into a training set, a validation set, and a test set. The training set contains 6471 images, the validation set contains 548 images, and the test set contains 1610 images. Figure 8 shows some images of the dataset VisDrone2021-DET.

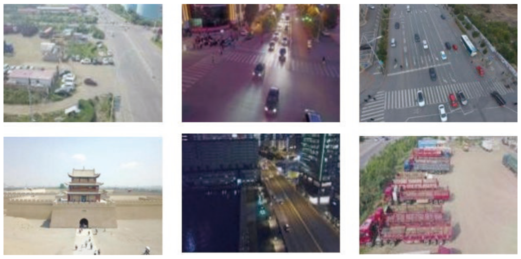


Figure 8 Selected images of the dataset VisDrone2021-DET.

2) Dataset CARPK

Figure 9 shows some images of the dataset CARPK. From the figures, we can see that some of the images are overexposed, with large differences between light and dark images. The targets are small and dense, but there are obvious boundaries between them. Some of the targets fail to enter the viewfinder frame in full and are truncated by the image boundaries. Target detection for the dataset CARPK requires to locate the targets accurately.

Figure 9 Selected images of the dataset CARPK.

2. Experimental setting and evaluation indicators

The proposed three lightweight networks are used for UAV aerial image target detection with batch size set to 32, and stochastic gradient descent (SGD) is chosen as the optimizer. The loss function is obtained by summing the confidence loss, classification loss and regression loss, and is used and trained on a single GPU. The training process is performed on the dataset CARPK, setting the epoch of training to 200, and on dataset VisDrone2021-DET, setting the epoch of training to 300. Other networks are trained according to the official training strategy.

The evaluation metrics include the number of parameters, weight, and mAP. The number of parameters is the number of trainable parameters of the network, and the more parameters, the heavier the network. The weight is the size of the weight file of the network, which is the most intuitive reflection of whether the network is lightweight or not, and the lightweight network is easier to deploy on mobile devices. mAP is mean average precision, each category has an AP value, and mAP is the average AP value of all categories. mAP is an important evaluation metric for the object detection task, which reflects the detection accuracy of the network. The higher the mAP value, the better the network detection effect.

3. Ablation experiments

In this paper, the benchmark network is YOLOv5n, and the ablation experiments are conducted on the datasets CARPK and VisDrone2021-DET, with the input image size set to 640×640. Firstly, the CSPGhostNeck residual structure and deconvolution are separately used in the benchmark network to verify the effects. Then, the CSPGhostNeck residual structure and deconvolution are simultaneously used in the benchmark network to verify the mutual exclusivity of the two.

1) Ablation experiments of CSPGhostNeck residual structure and deconvolution

Experiments are conducted on the dataset CARPK using the CSPGhostNeck residual structure and deconvolution, respectively. The experimental results are shown in Table 1 (M stands for 10^6). Baseline denotes the baseline network, Baseline+CSPGhostNeck denotes the first variant that the baseline network uses the CSPGhostNeck residual structure in the feature fusion stage, and Baseline+Deconv denotes the second variant that the deconvolution is introduced in the baseline network. We analyze the effect of CSPGhostNeck and deconvolution by comparing the performance of these two variants with Baseline.

Table 1 Results of the ablation experiments on the dataset CARPK

Network	Param (M)	Weight (MB)	mAP (%)
Baseline	1.76	3.83	82.0
Baseline+CSPGhostNeck	1.48	3.24	80.0
Baseline+Deconv	1.95	4.20	83.2

As shown in the table above, compared with Baseline, the parameter amount of Baseline+CSPGhostNeck is reduced by 0.28M, the weight by 0.59 MB, and the mAP by 2.0%. The CSPGhostNeck residual structure can reduce the number of parameters and the weight. The first reason is that CSPGhostNeck sends the input image into two branches and concatenates the results of two branches as the output, reducing the dimension of the intermediate feature maps. Moreover, the CSPGhostNeck contains multiple GhostNeck structures, which generate most feature maps by deep convolution, reducing the number of parameters and weight. However, the lightweight CSPGhostNeck residual structure degrades the performance of the detection network.

In comparison, Baseline+Deconv improves the mAP by 1.2%, while the number of parameters increases by 0.19M and the weight by 0.37 MB. Deconvolution improves the detection accuracy of the network, since the deconvolution allows the network to learn appropriate upsampling rules, so that the image retains more complete semantic information for better feature fusion, which in turns improves the detection accuracy. However, deconvolution requires the assistance of convolutional kernels, so it increases the number of parameters and weight of the network.

Experiments are conducted on the dataset VisDrone2021-DET using the CSPGhostNeck residual structure and deconvolution, respectively, and the experimental results are shown in Table 2.

Table 2 Results of the ablation experiments on the dataset VisDrone2021-DET

Network	Param (M)	Weight (MB)	mAP (%)
Baseline	1.77	3.85	13.0
Baseline+CSPGhostNeck	1.49	3.26	11.7
Baseline+Deconv	1.96	4.23	13.2

As shown in the table above, the number of parameters and weight of the network grow when the experiments are performed on dataset VisDrone2021-DET.

This is due to the fact that dataset CARPK has only one category, while dataset VisDrone2021-DET has 10, which leads to the difference between prediction heads and the increase of the number of parameters and weight of the network. The mAP on dataset VisDrone2021-DET has a significant decline compared to the mAP on dataset CARPK. There are many reasons of low mAP such as the difficulty of detection, the large number of categories, and complex background of dataset VisDrone2021-DET. In general, the same trend is observed on both the dataset VisDrone2021-DET and dataset CARPK using CSPGhostNeck residual structure and deconvolution, respectively. With CSPGhostNeck, the residual structure leads to a reduction in the number of parameters and weight, as well as a decrease in mAP. The introduction of deconvolution increases the number of parameters of the network and the weight as well as mAP.

The experimental results show that the CSPGhostNeck residual structure can effectively reduce the number of parameters and weight of the network at the expense of some accuracy. The deconvolution can effectively improve the detection accuracy of the network but increase a small number of parameters and weight. The CSPGhostNeck and deconvolution show the same characteristics in different datasets and have good applicability.

2) Mutual exclusion verification of CSPGhostNeck residual structure and deconvolution

Experiments are conducted on the dataset CARPK using both the CSPGhostNeck residual structure and deconvolution, and the experimental results are shown in Table 3.

Table 3 Experimental results of mutual exclusivity validation on the dataset CARPK

Network	Param (M)	Weight (MB)	mAP (%)
Baseline	1.76	3.83	82.0
Baseline+CSPGhostNeck+Deconv	1.66	3.61	80.5

As illustrated in Table 3, compared with Baseline, Baseline+CSPGhostNeck+Deconv has 0.1M fewer parameters, 0.22 MB less weight, and 1.5% lower mAP. As can be seen from Table 1, using the CSPGhostNeck residual structure alone reduces the parameter amount by 0.28M, the weight by 0.59 MB, but the mAP is reduced by 2.0%. Introducing deconvolution alone increases the parameter amount by 0.19M, the weight by 0.37 MB, but the mAP is improved by 1.2%. The differences between them are 0.09M decrease in parameter amount, 0.22 MB decrease in weight, and 0.8% decrease in mAP, which is similiar to the situation using both the CSPGhostNeck residual structure and deconvolution. Structurally, using CSPGhostNeck instead of CSPBottleNeck to fuse features reduces the number of parameters and weight and lowers mAP; using deconvolution instead of the nearest-neighbor interpolation to upsample increases the number of parameters and weight but improves mAP. To sum up, the effect of using two blocks approximately equals to the sum of the effects of using them separately.

Experiments are conducted on dataset VisDrone2021-DET, using both the CSPGhostNeck residual structure and deconvolution, and the experimental results are shown in Table 4.

Table 4 Experimental results of mutual exclusion verification on the dataset VisDrone2021-DET

Network	Param (M)	Weight (MB)	mAP (%)
Baseline	1.77	3.85	13.0
Baseline+CSPGhostNeck+ Deconv	1.67	3.64	12.2

In this table, the experimental results on the dataset VisDrone2021-DET have the same changes as those on the dataset CARPK. Compared with Baseline, using the CSPGhostNeck residual structure alone decreases the parameter amount by 0.28M, the weight by 0.59 MB, and the mAP by 1.3%. Introducing deconvolution alone increases the number of parameters by 0.19M, the weight by 0.38 MB, and the mAP by 0.2%. The adoption of CSPGhostNeck residual structure and deconvolution simultaneously decreases the number of parameters by 0.1M, the weight by 0.21 MB, and the mAP by 0.8%. The effect of using CSPGhostNeck residual structure and deconvolution simultaneously is approximately equal to the sum of the effect of using CSPGhostNeck residual structure and deconvolution alone.

The experimental results show that the CSPGhostNeck residual structure and deconvolution are not mutually exclusive, and the effects of using them together are approximately equal to the sum of the effects of using them alone. They have the same performance on different datasets and have good applicability.

4. Comparison experiments

Comparison experiments are conducted on the datasets CARPK and VisDrone2021-DET with the input image size set to 640 × 640. The three networks proposed in this paper are compared with several existing algorithms, including the YOLOv3-Tiny, YOLOv4-Tiny, YOLOX-Nano YOLOv5n, YOLO-L, YOLO-M, and the YOLO-S.

1) Comparison experiments on the dataset CARPK

Comparative experiments are conducted on the dataset CARPK, and the experimental results are shown in Table 5.

In the table, YOLOv5n is the benchmark network, and YOLOv3-Tiny, YOLOv4-Tiny, and YOLOX-Nano are involved into the comparison. From the table, we can see that the ordering of the number of parameters of these networks is different from the ordering of weight. The reason is that these methods use different encoding methods and generate different weight. In practical ap-

Table 5 Experimental results of comparison on the dataset CARPK

Network	Param (M)	Weight (MB)	mAP (%)
YOLOv3-Tiny	8.67	17.42	73.9
YOLOv4-Tiny	5.87	23.59	70.8
YOLOX-Nano	0.90	7.59	73.0
YOLOv5n	1.76	3.83	82.0
YOLO-L	1.74	3.80	82.8
YOLO-M	1.66	3.61	80.5
YOLO-S	1.48	3.24	80.0

plications, weight is a more important influencing factor.

Firstly, compared with YOLOv5n, YOLO-L reduces the parameters and weight and improves the mAP. YOLO-M and YOLO-S reduce the number of parameters, weight, and mAP. Specifically, YOLO-L reduces the number of parameters by 1.1%, the weight by 0.8%, and the mAP by 1.0%. According to the proof above, mAP is improved by using deconvolution to complete the upsampling operation. However, deconvolution uses a convolution kernel to offset the effect of deconvolution on network magnitude. YOLO-L uses Ghost CBS structure to adjust the dimensionality in addition with the maximum pooling operation for downsampling, which successfully reduces the number of parameters and weight. YOLO-M reduces the number of parameters by 5.7% and weight by 5.7%, while mAP is reduced by only 1.8%. YOLO-S reduces the number of parameters by 15.9% and weight by 15.4%, while mAP is reduced by only 2.4%. YOLO-S uses the CSPGhostNeck residual structure to reduce the parameter quantity and weight, but the processing capacity of CSPGhostNeck is lower than that of CSPBottleNeck, so the mAP decreases. YOLO-M uses both CSPGhostNeck residual structure and deconvolution, taking into account the accuracy while pursuing light weight, so the magnitude of YOLO-M is smaller than YOLO-L and the detection accuracy is higher than YOLO-S.

Then, YOLO-L, YOLO-M, and YOLO-S are compared with the rest of the networks. Among them, YOLOX-Nano has the smallest number of parameters with 0.90 MB, YOLO-S has the smallest weight with 3.24 MB, and YOLO-L has the largest mAP with 82.8%. To visually compare the performance of each network, a scatter plot is drawn with the weight and mAP as the horizontal and vertical coordinates, as shown in Figure 10.

In Figure 10, the closer the network is to the top left corner, the lighter the weight and the higher the mAP it uses. From the figure, we can see that YOLOv4-Tiny has the largest weight and the lowest mAP, which is the weakest comprehensive network; YOLOX-Nano has a lighter weight and the second lowest mAP, while YOLOv3-Tiny has a higher mAP and the second lowest weight, which are all weak comprehensive networks. The network with the strongest comprehensive performance is YOLO-L, YOLO-M and YOLO-S, which have high mAP and are near the upper left corner of the scatter plot.

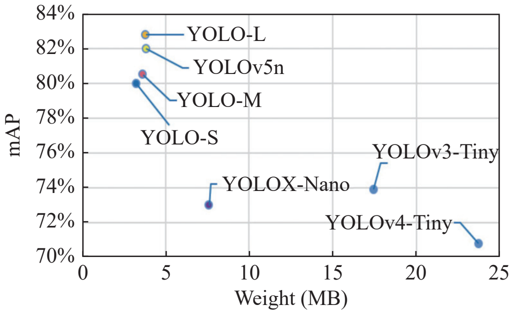


Figure 10 Comparison scatter plot on the dataset CARPK.

2) Comparison experiments on the dataset VisDrone-2021-DET

The comparison experiments are conducted on the dataset VisDrone2021-DET, and the experimental results are shown in Table 6.

Table 6 Comparative experimental results on the dataset VisDrone2021-DET

Network	Param (M)	Weight (MB)	mAP (%)
YOLOv3-Tiny	8.69	17.46	5.9
YOLOv4-Tiny	5.89	23.72	9.9
YOLOX-Nano	0.90	7.61	11.7
YOLOv5n	1.77	3.85	13.0
YOLO-L	1.75	3.82	13.1
YOLO-M	1.67	3.64	12.2
YOLO-S	1.49	3.26	11.7

Comparing with the benchmark network YOLOv5n, the experimental results on the dataset VisDrone2021-DET have the same trend as those on the dataset CARPK. YOLO-L reduces the number of parameters and weight and improves the mAP, while YOLO-M and YOLO-S sacrifice some of the mAP and significantly reduce the number of parameters and weight.

A comparison with the rest of the networks on dataset VisDrone2021-DET has some different experimental results from those on the dataset CARPK. Among them, YOLOX-Nano has the smallest number of parameters with 0.90M, YOLO-S has the smallest weight with 3.26 MB, and YOLO-L has the largest mAP with 13.1%. A scatter plot is drawn with the weight and mAP as horizontal and vertical coordinates, as shown in Figure 11.

From the figure, we can see that YOLOv3-Tiny has the lowest mAP and the second lowest weight, while YOLOv4-Tiny has the largest weight and the second lowest mAP. All of them perform weak comprehensively, YOLOX-Nano has a higher mAP, lighter weight, and stronger comprehensive performance. YOLO-L, YOLO-M, and YOLO-S have small weight and high mAP, near the upper left corner of the scatter plot, and outstanding comprehensive performance. On the two UAV aerial photography datasets, the comprehensive performances of YOLOv3-Tiny, YOLOv4-Tiny, and YOLOX-Nano are unstable, while YOLO-L, YOLO-M, and YOLO-S al-

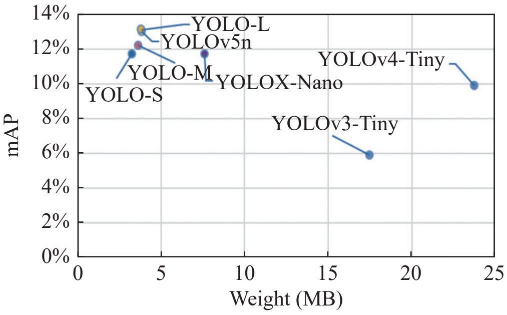
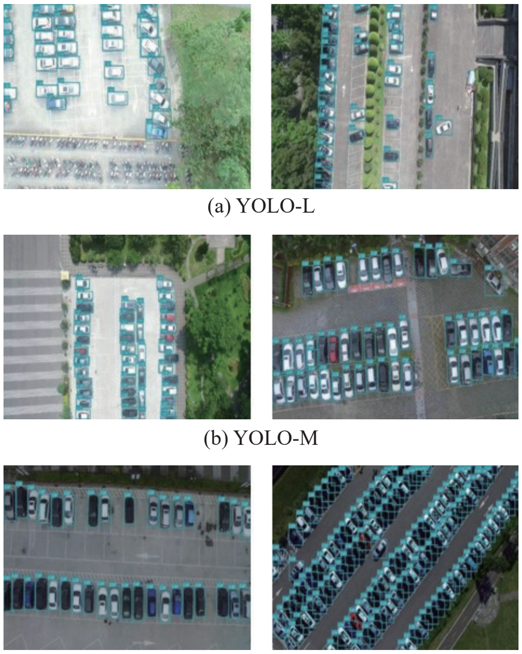


Figure 11 Comparison scatter plot on the dataset VisDrone2021-DET.

ways perform well in the upper left corner of the scatter plot and have better applicability.

The experimental results show that YOLO-L reduces the magnitude of the network and improves the detection accuracy, which is an excellent lightweight UAV aerial image target detection network; YOLO-M and YOLO-S can significantly reduce the number of parameters and weight of the network under the condition of sacrificing some detection accuracy and they are highly lightweight UAV aerial image detection networks. YOLO-L, YOLO-M, and YOLO-S have small weight and high mAPs, and perform stably in different UAV aerial image datasets, showing superiority in comparison with other networks. Figure 12 shows the visual detection results of YOLO-L, YOLO-M, and YOLO-S on the dataset CARPK.



(a) YOLO-L

(b) YOLO-M

(c) YOLO-S

Figure 12 Visual detection results of YOLO-L, YOLO-M, and YOLO-S on the dataset CARPK.

V. Conclusions

Based on the design idea and network structure of lightweight convolutional neural networks, this paper proposes three lightweight target detection networks for UAV aerial images: YOLO-L, YOLO-M and YOLO-S. The CSPGhostNeck is composed of CSPNet and GhostNeck, which can effectively reduce the number of parameters and weight of the detection network. The deconvolution enables the network to explore suitable upsampling rules during training process, so that small-sized feature maps retain more feature information, making feature fusion more effective and the network's detection accuracy higher. The experimental results show that the three designed lightweight networks all perform well in the target detection task of UAV aerial images.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61771319, 62076165, and 61871154), the Shenzhen Science and Technology Project (Grant Nos. JCYJ20180507182259896 and 20200826154022001), and other projects (Grant Nos. 2020KCXTD004 and WDZC20195500201).

References

[1] B. Rocke, A. Ruffell, and L. Donnelly, "Drone aerial imagery for the simulation of a neonate burial based on the geoforensic search strategy (GSS)," *Journal of Forensic Sciences*, vol. 66, no. 4, pp. 1506–1519, 2021.

[2] I. K. Hung, D. Unger, D. Kulhavy, *et al.*, "Positional precision analysis of orthomosaics derived from drone captured aerial imagery," *Drones*, vol. 3, no. 2, article no. 46, 2019.

[3] U. Andriolo, G. Gonçalves, N. Rangel-Buitrago, *et al.*, "Drones for litter mapping: An inter-operator concordance test in marking beached items on aerial images," *Marine Pollution Bulletin*, vol. 169, article no. 112542, 2021.

[4] H. Gupta and O. P. Verma, "Monitoring and surveillance of urban road traffic using low altitude drone images: A deep learning approach," *Multimedia Tools and Applications*, vol. 81, no. 14, pp. 19683–19703, 2022.

[5] Y. S. Li, S. F. Chen, W. H. Luo, *et al.*, "Hyperspectral image super-resolution based on spatial-spectral feature extraction network," *Chinese Journal of Electronics*, vol. 32, no. 3, pp. 415–428, 2023.

[6] A. Jain, R. Ramaprasad, P. Narang, *et al.*, "Ai-enabled object detection in UAVs: Challenges, design choices, and research directions," *IEEE Network*, vol. 35, no. 4, pp. 129–135, 2021.

[7] P. Mittal, R. Singh, and A. Sharma, "Deep learning-based object detection in low-altitude UAV datasets: A survey," *Image and Vision Computing*, vol. 104, article no. 104046, 2020.

[8] G. Y. Tian, J. R. Liu, H. Zhao, *et al.*, "Small object detection via dual inspection mechanism for UAV visual images," *Applied Intelligence*, vol. 52, no. 4, pp. 4244–4257, 2022.

[9] R. Walambe, A. Marathe, and K. Kotecha, "Multiscale object detection from drone imagery using ensemble transfer learning," *Drones*, vol. 5, no. 3, article no. 66, 2021.

[10] Z. K. Li, X. L. Liu, Y. Zhao, *et al.*, "A lightweight multi-scale aggregated model for detecting aerial images captured by UAVs," *Journal of Visual Communication and Image Representation*, vol. 77, article no. 103058, 2021.

[11] Y. Wang, "Survey on deep multi-modal data analytics: Collaboration, rivalry, and fusion," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 17, no. 1s, article no. 10, 2021.

[12] M. Sharma, M. Dhanaraj, S. Karnam, *et al.*, "YOLOrs: Object detection in multimodal remote sensing imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 1497–1508, 2021.

[13] Y. S. Li, H. J. Tang, W. X. Xie, *et al.*, "Multidimensional local binary pattern for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60 pp. 1–13, 2022.

[14] A. G. Howard, M. L. Zhu, B. Chen, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint*, arXiv: 1704.04861, 2017.

[15] M. Sandler, A. Howard, M. L. Zhu, *et al.*, "MobileNetV2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 4510–4520, 2018.

[16] A. Howard, M. Sandler, B. Chen, *et al.*, "Searching for mobileNetV3," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), pp. 1314–1324, 2019.

[17] X. Y. Zhang, X. Y. Zhou, M. X. Lin, *et al.*, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 6848–6856, 2018.

[18] N. N. Ma, X. Y. Zhang, H. T. Zheng, *et al.*, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *15th European Conference on Computer Vision*, Munich, Germany, pp. 122–138, 2018.

[19] Y. S. Li, L. D. Fan, and W. X. Xie, "TGSIFT: Robust SIFT descriptor based on tensor gradient for hyperspectral images," *Chinese Journal of Electronics*, vol. 29, no. 5, pp. 916–925, 2020.

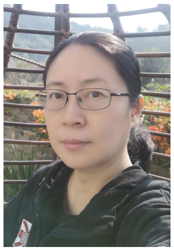
[20] K. Han, Y. H. Wang, Q. Tian, *et al.*, "GhostNet: More features from cheap operations," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, pp. 1577–1586, 2020.

[21] G. Jocher, K. Nishimura, T. Mineeva, *et al.*, "Yolov5," Available at: *https: //github. com/ultralytics/yolov5*, 2020.

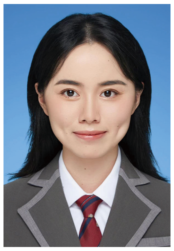
[22] Y. S. Li, T. Y. Guo, X. Liu, *et al.*, "Action status based novel relative feature representations for interaction recognition," *Chinese Journal of Electronics*, vol. 31, no. 1, pp. 168–180, 2022.

Yanshan LI is an Associate Professor with the ATR National Key Laboratory of Defense Technology, Shenzhen University, Shenzhen, China. He received the M.S. degree from Zhejiang University of Technology, Hangzhou, China, in 2005, and the Ph.D. degree from South China University of Technology, Guangzhou, China, in 2015. His research interests include computer vision, machine learning, and image analysis.
(Email: lys@szu.edu.cn)

Jiarong WANG received the M.S. degree from the College of Electronic and Information Engineering, Shenzhen University, Shenzhen, China, in 2022. His research interests include computer vision, deep learning, and image processing.
(Email: 2015130177@email.szu.edu.cn)



Kunhua ZHANG is an Associate Professor with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. She received the Ph.D. degree from the Chinese Academy of Sciences in 2003. Her research interests include computer vision and image analysis.
(Email: zhang_kh@szu.edu.cn)



Jiawei YI is currently pursuing the M.S. degree with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. Her research interests include computer vision, deep learning, and image processing.
(Email: 15007962908@163.com)

Miaomiao WEI is currently pursuing the M.S. degree with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. Her research interests include computer vision, deep learning, and image processing.
(Email: 2210434094@email.szu.edu.cn)

Lirong ZHENG received the B.E. degree from the College of Electronic and Information Engineering, Shenzhen University, Shenzhen, China, in 2019. She is currently pursuing the Ph.D. degree with Shenzhen University, Shenzhen, China. She is a member of the ATR National Key Laboratory of Defense Technology, Shenzhen University, Shenzhen, China. Her research interests include intelligent information processing, video rocessing, and pattern recognition.
(Email: zhenglirong2021@email.szu.edu.cn)

Weixin XIE received the degree from Xidian University, Xi'an, China. He was a Faculty Member with Xidian University in 1965. From 1981 to 1983, he was a Visiting Scholar at the University of Pennsylvania, USA. In 1989, he was a Visiting Professor with the University of Pennsylvania. He is currently working with Shenzhen University, Shenzhen, China. His research interests include intelligent information processing, fuzzy information processing, image processing, and pattern recognition.
(Email: wxxie@szu.edu.cn)