remote sensing

MDPI

Article

CBGS-YOLO: A Lightweight Network for Detecting Small Targets in Remote Sensing Images Based on a Double Attention Mechanism

Zhenyuan Wu [1,2], Di Wu [1], Ning Li [1,*], Wanru Chen [3], Jie Yuan [4], Xiangyue Yu [1] and Yongqiang Guo [1]

[1] Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China; wuzhenyuan22@mails.ucas.ac.cn (Z.W.); wudi@ciomp.ac.cn (D.W.); yuxiangyue@ciomp.ac.cn (X.Y.); guoyongqiang@ciomp.ac.cn (Y.G.)
[2] University of Chinese Academy of Sciences, Beijing 100049, China
[3] School of Civil Engineering and Environment, Hubei University of Technology, Wuhan 430068, China; ccgjs3888@163.com
[4] Software College of North University of China, North University of China, Taiyuan 030051, China; yj@st.nuc.edu.cn
* Correspondence: lining@ciomp.ac.cn; Tel.: +86-136-5430-8385

Abstract: With the continuous progress of remote sensing technology, the demand for means of detecting small targets in remote sensing images is escalating. The significance of detecting small targets in remote sensing images lies in enhancing the ability to identify small and elusive targets and the detection accuracy against complex backgrounds, holding significant application value in military reconnaissance, environmental monitoring, and disaster early-warning systems. Firstly, the minuteness of certain targets in relation to the entire image in which they occur, particularly when the camera is situated at a higher altitude, renders them difficult to detect. Secondly, the varying background and lighting conditions in remote sensing images further complicate the detection task. Conventional target detection methods are frequently incapable of addressing these complexities, resulting in a reduction in detection accuracy and an increase in false alarms. Hence, in this paper, we propose a lightweight remote-sensing image target detection network model, CBGS-YOLO, created by introducing the Ghost module to decrease the model parameters, applying the SPD-Conv module to optimize downsampling, and integrating the convolutional block attention module to enhance detection accuracy. The experimental outcomes demonstrate that CBGS-YOLO outperforms other models when applied to the DB_Licenta and USOD datasets, significantly enhancing detection performance for small targets. Compared with YOLOv9, this model can reduce the number of parameters from 7.10 M to 5.12 M, and the average precision (mAP) is effectively improved. The model strengthens the ability to identify small targets against complex backgrounds while maintaining lightweight properties and possesses remarkable application prospects and practical value.

Keywords: dim target detection; lightweight network; remote sensing

Academic Editor: Salah Bourennane

Received: 16 October 2024
Revised: 22 December 2024
Accepted: 27 December 2024
Published: 31 December 2024

Citation: Wu, Z.; Wu, D.; Li, N.; Chen, W.; Yuan, J.; Yu, X.; Guo, Y. CBGS-YOLO: A Lightweight Network for Detecting Small Targets in Remote Sensing Images Based on a Double Attention Mechanism. *Remote Sens.* **2025**, *17*, 109. https://doi.org/10.3390/rs17010109

1. Introduction

Optical remote sensing target detection technology relies on high-resolution aerial and satellite imagery obtained from visible-light sensors operating within the wavelength range of 0.38 to 0.76 microns, focusing on the identification and localization of critical objects such as aircraft, ships, and buildings in these images. As deep-learning techniques continue to advance alongside improvements in earth observation technologies, this field continues

to demonstrate significant potential for application across various domains. It provides essential foundational data for urban planning [1], land use management [2], and traffic oversight [3] while also playing a vital role in military surveillance [4] and emergency disaster-response efforts [5]. In particular, when addressing the difficulties associated with small-target detection, optical remote-sensing-image target-detection technology leverages its unique strengths to establish a solid technical framework that enables more accurate and efficient monitoring through remote sensing methods. High-altitude images often contain many small targets with pixel dimensions limited to between 10×10 and 20×20 pixels, complicating effective detection via traditional networks. Additionally, these images can be adversely affected by cloud cover during acquisition, resulting in compromised image quality and obscured feature details. Therefore, investigating deep-learning approaches in order to tackle the challenges posed by small-target detection in remote sensing is crucially important and offers substantial practical benefits.

As a key field of remote sensing, remote-sensing-based small-target detection has led to some achievements, but it still faces multiple challenges. First of all, due to the minuteness of such targets, they occupy a limited number of pixels in a high-resolution image, resulting in scarce feature information, which is easily altered by background noise or confused with similar objects, thus increasing the difficulty of accurate detection [6]. Secondly, remote sensing images often cover complex and changeable surface landscapes, such as urban buildings, dense forests, and vast water bodies; these backgrounds may contain textures or colors similar to those of the target, resulting in significant interference with detection [7]. Moreover, remote sensing imaging conditions are complex and changeable, being affected by different lighting conditions, seasonal changes, weather conditions, etc., which will affect image quality and, thus, the efficacy of detecting small targets. Therefore, determining how to accurately and efficiently identify small targets within complex backgrounds is still an urgent problem in the field of remote-sensing-based small-target detection.

In conclusion, the challenges faced by remote sensing object detection mainly include the following two aspects:

- Complex backgrounds in remote sensing imagery may hinder target detection accuracy.
- The computational complexity of remote-sensing-based object detection networks results in slow inferencing, making it difficult to meet real-time requirements.

As a result of the previously mentioned challenges, including the intricacies of network computing, inadequate model performance in handling complex backgrounds, and difficulties in detecting small targets, in this study, we focus on two key areas for improvement: First, acknowledging the unique characteristics of remote sensing images, we enhance the YOLOv5 framework to significantly improve both precision and robustness in recognizing small targets within these images. The aim of this optimization is to better address the complexities and variability inherent in remote sensing environments. We have chosen YOLOv5 as our reference framework due to its more efficient parameter count compared to the latest version, YOLOv10, while still being able to achieve a high level of accuracy for small-target-detection tasks. Second, we aim to develop lightweight model architectures that balance high recognition accuracy with reduced parameter size and computational complexity, ensuring that our model meets real-time processing demands and operates efficiently during practical application.

The organization of this article is succinctly presented as follows: In the first section, we offer an overview of the applications of remote sensing technology and conduct an in-depth analysis of the challenges encountered in detecting pedestrians in drone-based remote sensing images. The second section reviews the evolution of object detection technology, with particular emphasis on the development of small-target detection and lightweight networks. In the third section, we describe the construction of a lightweight detection

network based on YOLOv5, and its innovative enhancements are elaborated upon. In the fourth section, we describe experiments carried out on two datasets to comprehensively validate the efficacy of the proposed improvement method. In the fifth section, an in-depth discussion of the experimental results is provided alongside the results of ablation experiments performed to verify the reliability of the improved modules. Finally, the entire work is summarized in the last section, and potential future research directions are given.

2. Related Work

2.1. Development of Object Detection

The field of object detection has made a technological leap from traditional machine learning to deep learning. Early on, researchers relied on artificial features (such as SIFT [8], HOG [9], and Haar features [10]) and classical classifiers, which performed well when applied to simple settings but had limited accuracy and adaptability in the face of complex environments. Similarly, traditional target detection methods for remote sensing images (such as threshold segmentation [11] and edge detection [12]) are also limited by manual features, and it is difficult to meet the high precision requirements. Traditional algorithms often include candidate region generation (such as sliding windows), feature extraction, and classifier classification, where non-maximum suppression (NMS) [13] is used to eliminate redundancy, but a significant amount of manual adjustment is required, limiting its applications. Although traditional methods still have value in specific scenarios, deep learning methods have become the new mainstay in object detection due to their high accuracy and robustness in processing complex images.

With the rapid development of deep learning technology, object detection algorithms based on deep learning can now directly learn feature representations from rich annotated data by relying on deep neural networks, greatly improving the accuracy and efficiency of object detection. These algorithms are mainly divided into two categories: two-stage detectors, which first generate potential target candidate regions and then classify these regions and adjust the regression of bounding boxes, and single-stage detectors, which synchronously predict the category and bounding box of the target in a single step, endowing them with greater detection speed. In the context of the rapid advancement of deep-learning technology, target detection algorithms that leverage deep learning can directly acquire feature representations from extensive annotated datasets, thereby greatly improving both the accuracy and efficiency of target detection. These algorithms are generally classified into two main categories: two-stage detectors, which begin by generating potential candidate regions for targets and then proceed to classify and refine the bounding box regression for these regions, and one-stage detectors, which simultaneously predict both the category and bounding box of a target in a single step, thus facilitating greater detection speed.

In the evolution of two-stage object detection algorithms, GIRSHICK et al. [14] shattered the constraints of traditional algorithms by introducing R-CNN. They initially selected candidate regions, subsequently extracting features through convolutional neural networks (CNNs), classifying them with support vector machines (SVMs), and obtaining location information by means of a fully connected network. To tackle the issue in which R-CNN consumes a considerable amount of time in processing overlapping regions, He et al. [15] proposed SPP-Net, which introduced the spatial pyramid pooling (SPP) layer, enabling the entire image to be input into the CNN and thereby generating a fixed feature representation for any region. Subsequently, Ren et al. further advanced Fast R-CNN [16] on the basis of R-CNN and SPP-Net. They substituted AlexNet with VGG-16 and employed region-of-interest (RoI) pooling in place of SPP. Simultaneously, they transformed the traditional binary classification task into a softmax function, inte-

grated all the steps, and notably enhanced the speed and accuracy. Nevertheless, Fast R-CNN is still restricted by the speed of candidate region extraction. To surmount this limitation, Ren et al. [17] proposed Faster R-CNN, replacing the selective search with a region proposal network (RPN). The RPN is positioned after the convolutional layer, achieving feature sharing and thereby significantly boosting training efficiency. Based on this, DOLLÁR P et al. optimized Faster R-CNN by designing Mask R-CNN [18], which combines the functions of instance segmentation and object detection. Mask R-CNN adds a mask branch to the original boundary box recognition branch, which is utilized for predicting object masks. In this manner, when detecting objects in an image, Mask R-CNN can generate high-quality segmentation masks for each instance concurrently, further enhancing detection accuracy. To address the challenge of IoU threshold selection in the R-CNN series of algorithms, Cai et al. [19] proposed Cascade R-CNN. This approach can realize the progressive refinement of target selection and localization by cascading three detection heads trained for different IoU threshold values. The input of each detection head stems from the output of the previous detection head.

In the field of single-stage object detection, the YOLO algorithm series, first introduced by Redmon et al. [20] in a groundbreaking endeavor, has gained recognition for its remarkable real-time detection speed and continuously improving accuracy. YOLOv2 and YOLOv3 [21,22] significantly boosted detection performance through the adoption of efficient feature extraction methods and integrated training strategies. Subsequently, Bochkovskiy et al. [23] launched YOLOv4, which further enhanced the detection capabilities. The team at Ultralytics LLC developed YOLOv5 [24] by incorporating adaptive technology and a focus structure to refine YOLOv4's performance. Li et al. [25] introduced YOLOv6, utilizing an innovative backbone network called EfficientNet along with advanced techniques. Wang et al. [26] released YOLOv7 using model reparameterization, among other approaches. By 2024, versions such as YOLOv9 [27], YOLOv10 [28], and their subsequent iterations were expected to continue establishing new standards in terms of accuracy and speed in detection tasks. Presently, the YOLO series is widely utilized across various sectors, including robotics, autonomous vehicles, and video surveillance systems, driving significant progress in these areas. However, SSD is another foundational model in single-stage object detection. Proposed by Liu et al. [29], SSD employs VGG-16 as its backbone architecture while achieving multi-scale predictions through multiple layers of progressively smaller convolutional features. It utilizes small convolutional kernels to effectively determine both category classifications and offsets for default bounding boxes. To enhance small-target-detection capabilities further, algorithms like DSSD [30] and FSSD [31], among others, have been developed. Additionally, Microsoft Research unveiled the Swin Transformer [32] in 2021, representing an innovative visual framework characterized by a window-sliding mechanism that computes self-attention locally while improving information exchange between neighboring windows via displacement techniques. Meanwhile, DETR [33] was introduced, representing a transformer-based target detection framework that eliminates traditional anchor boxes alongside complex post-processing procedures such as non-maximum suppression.

The field of detecting small targets in drone-acquired remote sensing imagery is still developing. Researchers primarily depend on manually crafted features and traditional classifiers to perform target detection tasks. Rozantsev et al. [34] introduced a regression-based method for stabilizing the motion of local image patches. It enabled effective detection of temporal and spatial image cubes, surpassing the leading technologies available at that time. Sun et al. [35] proposed a new detection framework utilizing the spatial sparse coding bag-of-words (SSCBOW) model. Specifically, after selecting processing units via a sliding window approach and extracting their features, they implemented an innovative

spatial mapping strategy to encode geometric information effectively. This strategy not only illustrates the relative positions among various components of the targets but also has the ability to accommodate rotational changes. Deng et al. [36] recommended a technique based on weighted local differential measurement (WLDM) designed to enhance targets while minimizing background noise in complex cloudy environments, achieving accurate detection and segmentation for small targets. Conventional target detection methods depend on the precision of manually designed features, while those based on deep learning can automatically extract distinguishing features, making them increasingly preferred by researchers and widely utilized for detecting targets in drone-acquired aerial imagery. Zhang et al. [37] introduced Drone-YOLO by implementing a three-layer PAFPN architecture and creating a detection head that leverages large-scale feature maps. It significantly improved the ability to detect small targets in drone-based remote sensing images. Tan et al. [38] re-sampled feature images using dilated convolution techniques and incorporated an ultra-light quantum spatial attention mechanism (ULSAM) that produced unique attention feature maps for each subspace of the feature map, thus facilitating an accurate representation of multi-scale features.

2.2. Lightweight Network

In recent years, numerous researchers have engaged in comprehensive studies within the field of model lightweighting, which includes approaches such as model compression [39], quantization [40], knowledge distillation [41], and network pruning [42]. These techniques are widely applied to various neural networks to reduce both the number of parameters and the computational demand, aiming to improve the efficiency of classification and prediction tasks. With the rapid evolution of mobile devices and embedded systems, lightweighting technologies for models have become essential for deploying neural network architectures on these platforms. Depending on whether pre-training is involved, methods for model lightweighting can be classified into two main categories: the first category involves complex model compression techniques that focus primarily on compressing pre-trained models to decrease their size and streamline computation, while the second category consists of lightweight design strategies that bypass pre-training altogether to create compact and efficient neural networks directly.

The aim of complex model compression is to simplify neural networks by eliminating redundancies to reduce data volume and streamline structures, facilitating deployment. The commonly employed methods include parameter pruning, quantization, low-rank decomposition, knowledge distillation, and hybrid approaches. These methods reduce computational and storage requirements while maintaining performance and accelerating training speed. For instance, Hanson et al. [43] pruned parameters based on their magnitudes, and Courbariaux et al. [44] proposed Binary Connect, which scales weights to 1 or -1. Moreover, BinaryNet [45] further binarizes activation values and weights. Jaderberg et al. [46] decomposed k$\times$k convolutional kernels into rank-1 kernels, and Shen et al. [47] employed selective learning schemes to enable the student model to adaptively learn from the teacher model.

Lightweight model design includes both manual optimization and automated approaches, with the aim being to create compact models. In terms of manual optimization, spatial convolution design refines the convolution method and thus reduces the model size and computational requirements, with spatial-separable and depth-separable convolutions serving as fundamental techniques. The MobileNet architecture introduced by Howard et al. [48]; its optimized iteration, MobileNetV2; as well as ShuffleNet, proposed by Zhang et al. [49], all implement depth-separable convolution while incorporating innovative methods. However, lightweight models must maintain certain network

dimensions to prevent information loss and degradations in accuracy. To tackle this challenge, shift convolution design was developed, in which a more compact model was constructed by replacing conventional convolution operations. ShiftNet [50] exemplifies this area; however, determining the allocation of shift amounts presents training challenges. Jeon's As-ResNet [51] employs active shift learning to determine the shift amount and incorporates an active shift layer that addresses allocation issues related to shifts. Nonetheless, the reliance on memory operations for shifting affects the overall efficiency during execution. Therefore, hardware support is essential for effectively implementing shift convolution models. While these models excel at reducing computation load and parameters, they still face challenges such as prolonged training times and diminished operational efficiency in real-world applications.

3. Materials and Methods

3.1. The Framework of CBGS-YOLO Network

To address the challenges posed by a high number of parameters and suboptimal detection accuracy in small-target detection models for drone-based remote sensing images, in this paper, we introduce a lightweight model named CBGS-YOLO, where CBGS stands for Convolutional Block Attention Model, GhostNet, and SPD-Conv. The proposed model effectively integrates these components and represents an enhancement compared to the original YOLOv5 framework, with the aim being to improve its efficacy in detecting small targets in drone imagery. The architecture of CBGS-YOLO is illustrated in Figure 1.

In the optimization of the YOLOv5 framework, we adhered to a dual-design philosophy focused on enhancing efficiency and preserving accuracy, making meticulous refinements to the original architecture. In the backbone section, we substituted the conventional CBS module with the GhostCBS module to achieve model lightweighting through parameter reduction (refer to the left side of Figure 2). In the neck section, we innovatively integrated the GC_Neck module, based on GhostConv, to further refine the network structure and enhance information transmission efficiency, thereby maintaining accuracy while accelerating the operational speed. Additionally, we incorporated the Space_to_depth step following the GhostCBS module in the backbone to strengthen the network's feature extraction capabilities, enabling the model to more effectively capture critical features within images.

To further enhance the model's performance, we integrated the Convolutional Block Attention Model (CBAM) into each C3 module, with the objective of improving the model's detection accuracy for small targets (refer to the right half of Figure 2). Additionally, we incorporated the SPD-Conv module into the concat module in the neck section to augment the model's ability to handle fine details. These enhancements not only underscore our commitment to both performance and efficiency but also exemplify our design philosophy of achieving faster model execution and superior feature extraction while maintaining the high precision of the YOLOv5 framework through lightweight processing, module optimization, and feature enhancement.

In the model optimization process, we integrated the backbone section and combined the adopted GhostCBS, Space_to_depth, and C3_CBAM modules into an efficient composite module. The experimental results demonstrated that using three groups of this composite module could enhance detection accuracy while reducing the model's complexity. After being processed via the SPFF module, the data were transmitted to the neck section. In the neck, we adopted a modular design and integrated the GhostConv, UpSample, concat, GC_Neck, and C3_CBAM modules. We selected two groups to optimize performance. Subsequently, the data were passed to the subsequent feature fusion module, where the feature maps concatenated were the outputs from the previous module (i.e., the

GhostConv convolution), which were processed by the SPD module before concatenation. The primary function of the SPD module was to reduce the channel dimensions of the downsampled feature maps while retaining the information, thereby dispersing the feature map information to reduce computational load. Finally, the detection content was output through the output module.

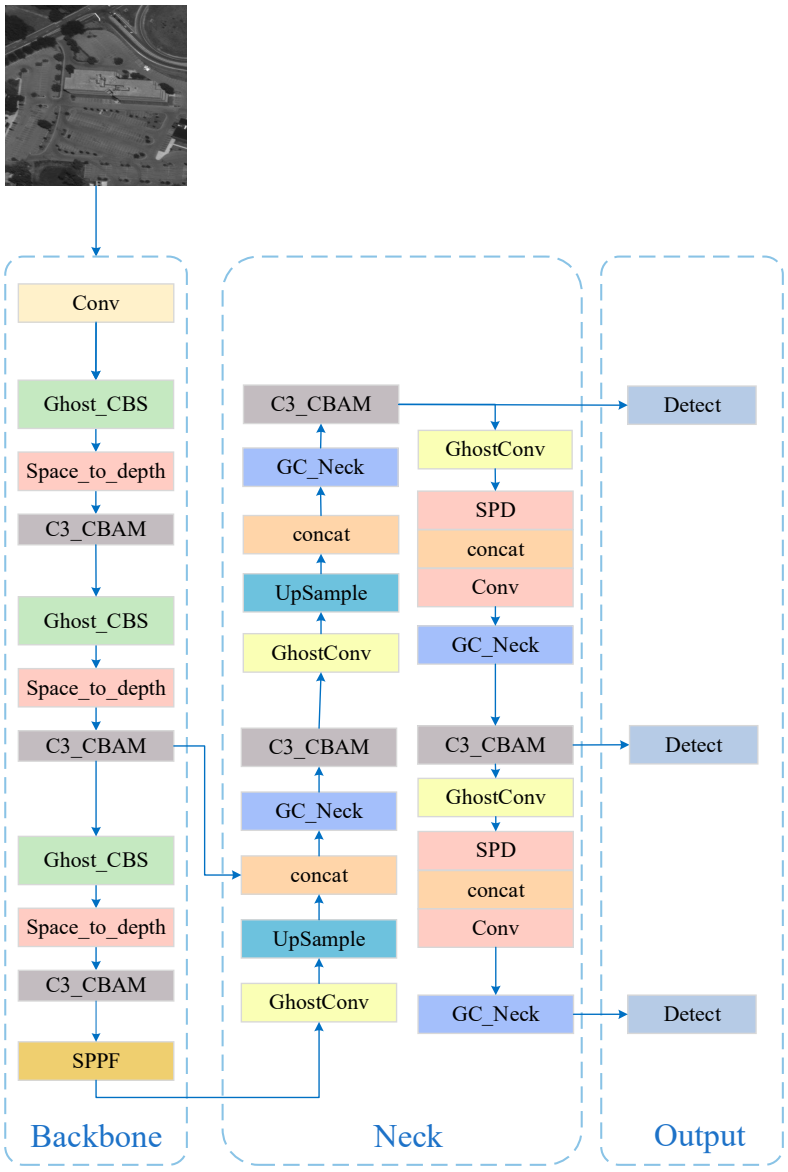


Figure 1. Overall structure of the CBGS-YOLO model. We replaced the convolutional structure in the original YOLOv5 backbone with Ghost_CBS and added GhostConv and GC_Neck modules in the neck. We used the SPD-Conv module for downsampling operations and added the Convolutional Block Attention Model mechanism to improve detection accuracy.

3.2. Convolutional Block Attention Model

To enhance the extraction of contextual features pertaining to small infrared targets, we incorporated the CBAM attention module [52] into this network (Figure 3). This module comprises both a channel attention module (CAM) and a spatial attention module (SAM). The CAM dynamically modulates the contribution of each channel to the output, while the SAM effectively identifies the precise locations of targets within the feature map. The integration of these two components enables the network to focus more acutely on

positional information and intricate edge details associated with small targets in each feature map.

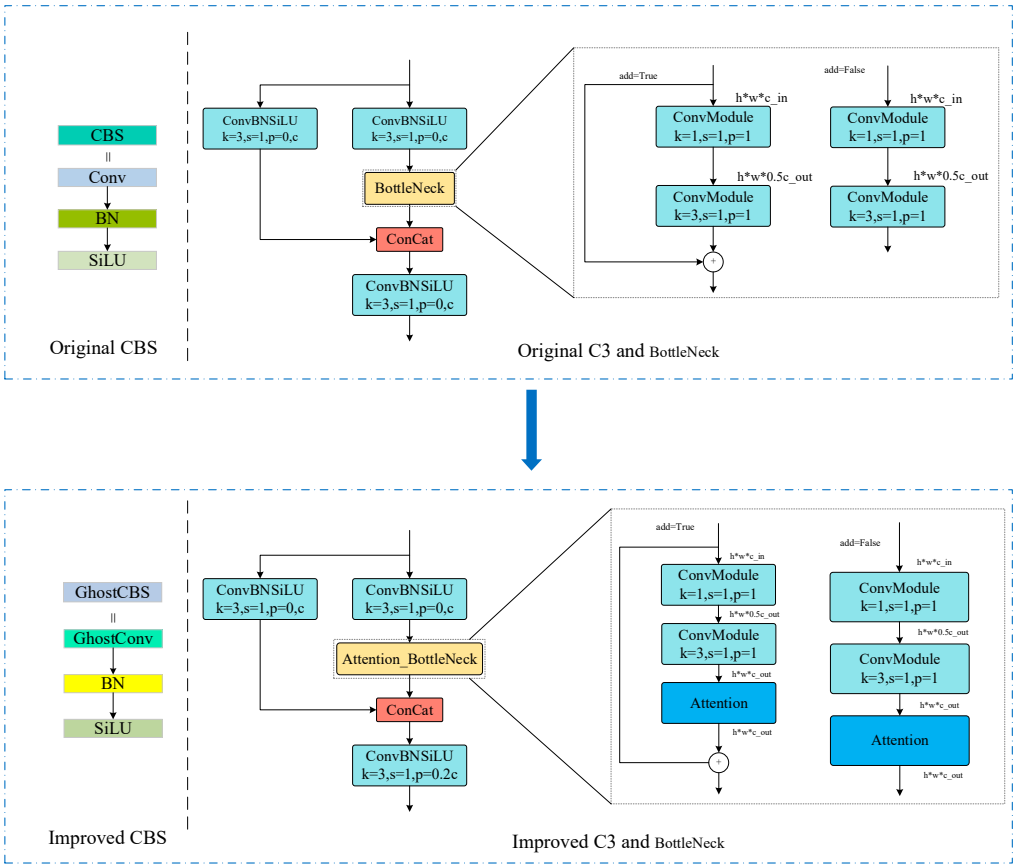


Figure 2. The replaced parts and a comparison with the original YOLOv5 modules. The upper half of the figure shows the original modules of YOLOv5, while the lower half shows the replacement units we used. This modification effectively reduced the number of model parameters and improved the speed and detection accuracy of model training.

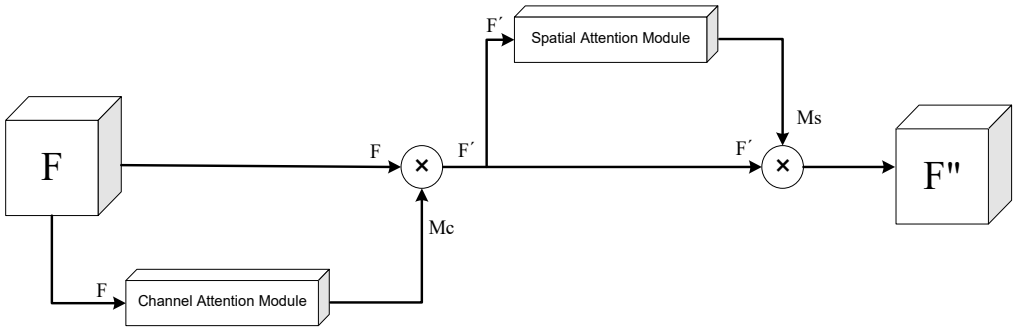


Figure 3. Convolutional Block Attention Model (CBAM). CBAM consists of two key components: the C-channel channel attention module and the S-channel spatial attention module. These two modules can be embedded into different layers of a CNN to enhance feature representation.

As shown in the figure above, the model consists of an input, a channel attention module, a spatial attention module, and an output. The input feature is $F = R^{C*H*W}$, followed by a one-dimensional convolutional module $M_c = R^{C*1*1}$ for channel attention. The convolution result is multiplied by the original image, and the output of CAM is used as the input for the spatial attention module's two-dimensional convolution, $M_s = R^{1*H*W}$. Finally, the output result is multiplied by the original image. The above content can be summarized by Equations (1) and (2), as follows:

$$F' = M_c(\mathrm{F}) \bigotimes \mathrm{F} \tag{1}$$

$$F'' = M_c(F') \bigotimes F' \tag{2}$$

3.2.1. Channel Attention Module

A flowchart of the CAM is shown in Figure 4. The working principle of the channel attention module is as follows: The input feature map is passed through two parallel MaxPool and AvgPool layers, which reduce the feature map from C×H×W to C×1×1 in size. Then, it is passed through the Share MLP module, where it first compresses the channel number to 1/r of the original size, then expands it back to the original size, and finally passes through the ReLU activation function to obtain two activated results. The two output results are added elementwise and then passed through a sigmoid activation function to obtain the output result of channel attention. Finally, this output result is multiplied by the original image so that its dimensions are C×H×W again.

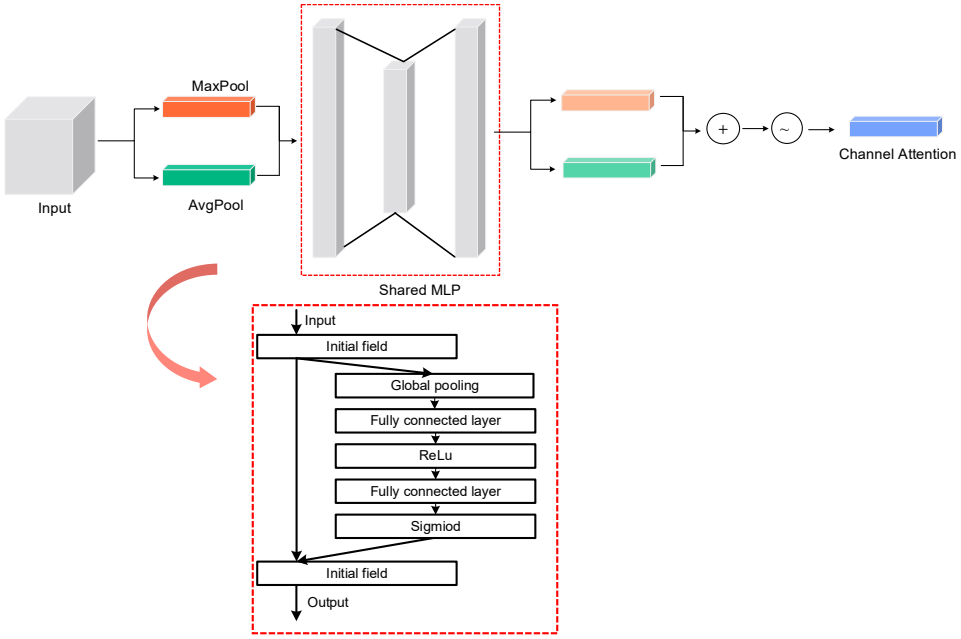


Figure 4. Channel attention module is used to handle the allocation relationship of feature map channels, while the attention allocation on two dimensions enhances the effect of attention mechanism on model performance.

A key feature of the channel attention module is that the channel dimensions remain unchanged while it compresses the spatial dimension. This module focuses on meaningful information in the input image. Its module structure is shown in Figure 4, and the channel attention formula ($Mc(F)$) is shown in Equation (3):

$$M_c(F) = \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \tag{3}$$

3.2.2. Spatial Attention Module

A flowchart of the SAM is shown in Figure 5. Its working principle is as follows: By passing the input feature map through two parallel MaxPool and AvgPool layers, the feature map is reduced from C×H×W to C×1×1 in size. Then, it is passed through the Share MLP module, where it first compresses the number of channels to 1/r of the original, then expands it back to the original number of channels, and finally passes through the ReLU activation function to obtain two activated results. The two output results are added elementwise and then passed through a sigmoid activation function to obtain the output

result of channel attention. Finally, the output result is multiplied by the original image so that its dimensions are C*H*W again.

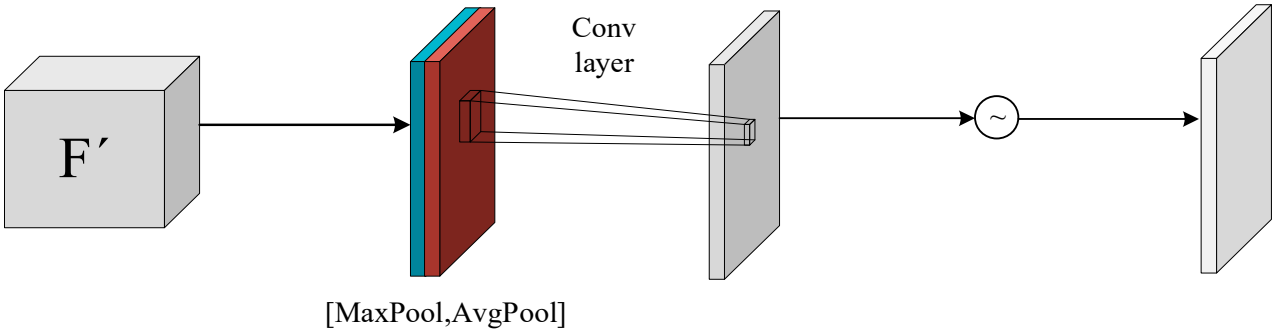


Figure 5. Spatial attention module enables neural networks to pay more attention to the pixel regions in an image that are crucial for classification, while ignoring less important regions.

The spatial attention module is characterized by the ability to preserve spatial dimensions while compressing channel dimensions. This module emphasizes the target's location information. The spatial attention formula ($Ms(F)$) can be expressed using Equation (4), as follows:

$$M_s(F) = \sigma\left(f^{7*7}([AvgPool(F\ ); (MaxPool(F)])\right) \tag{4}$$

3.3. Ghost

GhostNet [53] represents a lightweight convolutional neural network (CNN) architecture that has been specifically designed to improve both the efficiency and accuracy of tasks related to image classification. This framework effectively reduces computational requirements and minimizes the total number of parameters by utilizing cost-effective operational algorithms. The process followed by the Ghost module is as follows: First, the input data are processed through a conventional convolution operation, yielding a collection of intrinsic feature maps referred to as Y_1. Next, group convolution methods are applied to generate a set of highly correlated but slightly varied Ghost feature maps, $y_{i,j}$, derived from each channel feature map, $y_1^{'}$, using a designated transformation, $\Phi_{i,j}$. In conclusion, these intrinsic and Ghost feature maps are combined via an identity connection to create the final output feature map. This approach not only ensures efficient extraction of features, but also significantly alleviates computational complexity while decreasing parameter size in the network. The structure of the Ghost module is illustrated in Figure 6 below.

The operational mechanism of Ghost can be summarized as follows: Given an input feature map F in $\mathbb{R}^{c\times w\times h}$ (where c represents the number of channels, w denotes width, and h signifies height), passing it through a convolutional kernel with dimensions of n×k×k results in a feature map F′ in $\mathbb{R}^{C^{'}\times w^{'}\times h^{'}}$ At this stage, the parameter count and computational complexity associated with the standard convolution are as follows:

$$cost_1 = h^{'} \times w^{'} \times n \times c \times k \times k \tag{5}$$

$$cost_2 = \frac{n}{s}\cdot h^{'}\cdot w^{'}\cdot c\cdot k\cdot k + (s-1)\cdot\frac{n}{s}\cdot h^{'}\cdot w^{'}\cdot d\cdot d \tag{6}$$

And the number of participants and the computational load in Ghost are:

$$participants_1 = n \times c \times k \times k \tag{7}$$

$$participants_2 = \frac{n}{s}\cdot c\cdot k\cdot k + (s-1)\cdot\frac{n}{s}\cdot d\cdot d \tag{8}$$

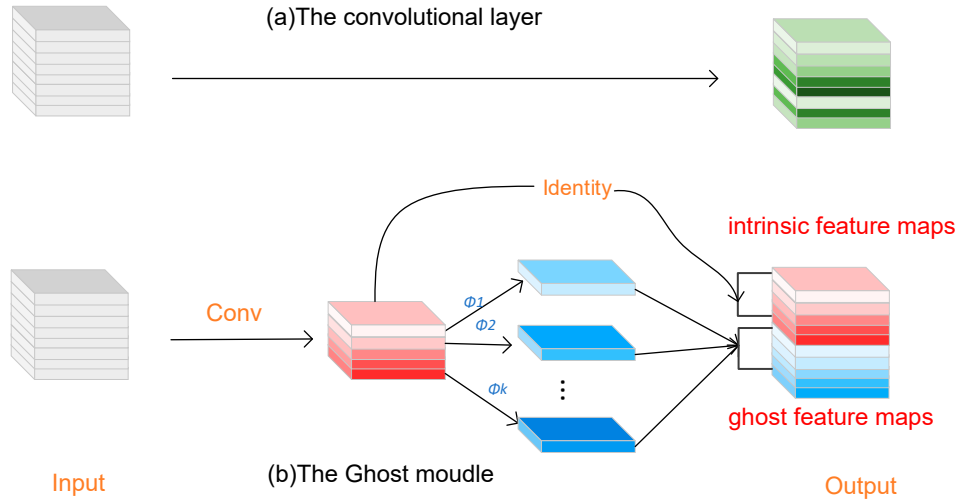


Figure 6. This comparison chart illustrates the differences between regular convolution and Ghost convolution. Regular convolution directly applies convolution to the input feature map, resulting in an output feature map characterized by high computational complexity and a substantial number of parameters. In contrast, Ghost convolution is achieved through three distinct steps: regular convolution, Ghost generation, and feature map fusion. Initially, Ghost convolution performs standard convolution to produce a limited set of intrinsic feature maps; subsequently, it generates additional similar feature maps via a series of low-cost operations, thereby significantly reducing both computational complexity and parameter count.

In Equations (5) and (6), $cost_i$ refers to the computational effort required to execute a convolution operation. The variables h and w represent the height and width of the input feature map, respectively, while c denotes the number of channels. Likewise, h' and w' indicate the height and width of the output feature map, with n representing the number of output feature channels. Finally, k indicates the dimensions of the convolution kernel.

In Equations (7) and (8), $participants_j$ denotes the computational demands for both conventional convolution and Ghost convolution. Here, d·d indicates the size of the convolution kernel used in linear operations, and s represents the number of linear transformations with s << c. The term n/s refers to the output channel count from the initial transformation, while s-1 accounts for the fact that identity mapping does not require computation; however, it is still included in the second transformation process, allowing the Ghost module to reduce the computational load. In Formula (9), r_s signifies the ratio of computational complexity with respect to conventional convolution and GhostConv convolution, whereas in Formula (10), r_c illustrates the ratio of parameter counts between these two types of convolutions.

$$r_s = \frac{h^{'} \cdot w^{'} \cdot n \cdot c \cdot k \cdot k}{\frac{n}{s} \cdot h^{'} \cdot w^{'} \cdot c \cdot k \cdot k + (s-1) \cdot \frac{n}{s} \cdot h^{'} \cdot w^{'} \cdot d \cdot d} \approx s \tag{9}$$

$$r_c = \frac{n \cdot c \cdot k \cdot k}{\frac{n}{s} \cdot c \cdot k \cdot k + (s-1) \cdot \frac{n}{s} \cdot d \cdot d} \approx \frac{s \cdot c}{s + c - 1} \approx s \tag{10}$$

As shown in Figure 7, the basic module (Ghost bottleneck) of a GhostNet consists of two Ghost modules. In a G-bneck, the first Ghost module is used to increase the number of channels (serving as the expansion layer), specifying the ratio of output to input channels as the expansion ratio. The second Ghost module reduces the number of channels to match the channel number of the shortcut branch. When the stride is 2, a depth-wise convolution layer with a stride of 2 is added between the two Ghost modules.

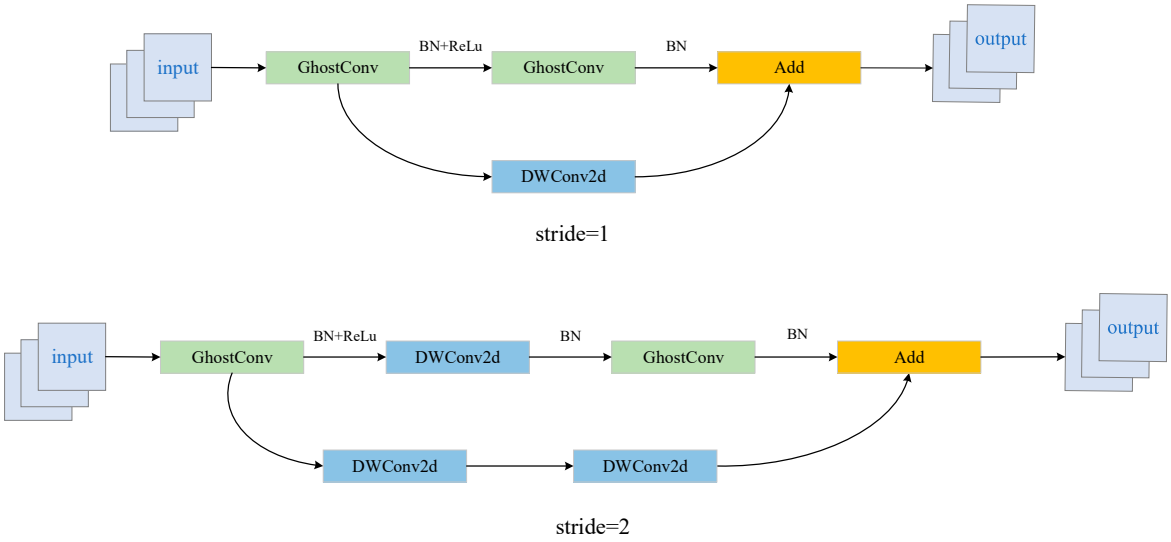


Figure 7. There are two Ghost modules in a G-bneck architecture, with the first one used to increase the number of channels (expansion layer), specifying the ratio of output to input channels as the expansion ratio. The second Ghost module reduces the number of channels to match the channel number of the shortcut branch. When the stride is 2, a depth convolution layer with a stride of 2 is added between the two Ghost modules.

However, if GhostConv is used at all stages of the model, the network layer of the model will be deeper, which will intensify the resistance to data flow and significantly increase the number of parameters and inference time. When these feature maps reach the neck, they will have become thin and tall (the channel dimension reaches its maximum, while the width and height dimensions reach their minimums) and no longer need to be transformed. Therefore, a better option is to use a cross-level network module based on GhostConv—GC-neck, at this stage. At this stage, using this module to process the stitched feature maps provides a solution that is just right: there is less redundant and repetitive information and no need for compression, and the computational speed is improved. The module reduces the complexity of computation and the network structure while maintaining sufficient accuracy. Figure 8 depicts the structure of the GC-neck.

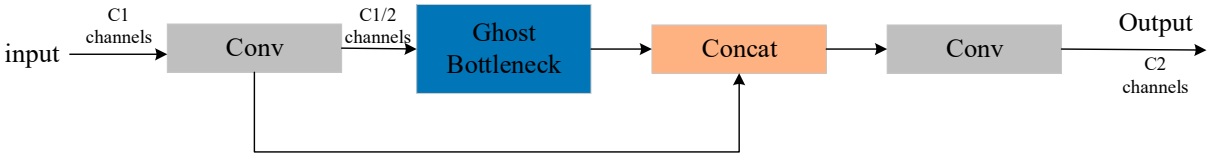


Figure 8. One of the three design options provided by GC-neck. This design is simple and direct in structure and the inference speed is fast, making it well-suited for embedded systems.

3.4. SPDConv

Conventional object detection networks face significant challenges in accurately recognizing small objects, largely due to their fundamental design limitations. Essential convolutional operations, such as stride convolution and pooling, inevitably result in a reduction in spatial resolution during the downsampling process, leading to a considerable loss of vital feature information. Furthermore, small objects often have a limited spatial presence, making it difficult for them to correspond with the fixed receptive fields defined by traditional networks. This misalignment restricts the ability to capture contextual information effectively and diminishes the network's ability to thoroughly comprehend and precisely locate small objects. Additionally, traditional networks' constraints in processing spatial context—combined with issues related to resolution degradation—impede their

capacity to adequately learn and extract unique features associated with small objects. Consequently, these issues complicate accurate differentiation between small objects and larger ones or background elements in complex environments, thereby negatively impacting overall detection performance.

The core principle of SPD-Conv [54] (spatial-to-depth convolution) is to improve the performance of traditional convolutional neural networks (CNNs) in handling small objects and low-resolution images. This aim is achieved through several essential mechanisms: 1. Alternative design: SPD-Conv is designed to replace stride convolutional layers and pooling layers in standard CNN architectures, as these elements are prone to losing fine details when processing low-resolution images or small objects. 2. The SPD (space-to-depth) conversion layer: The function of the SPD layer is to reduce the channel dimensions of the feature map while ensuring that vital information remains preserved, thus addressing the common issue of information loss experienced when using conventional methods. 3. A stride-free convolutional layer: After implementing the SPD layer, SPD-Conv introduces a convolutional layer with a stride set to one (i.e., no strides). This setup aids in refining features while simultaneously decreasing channel counts using trainable parameters. The unique structure of SPD-Conv effectively avoids information loss linked with stride convolutions and pooling operations, thereby maintaining detailed image characteristics and significantly enhancing feature representation abilities. These qualities allow SPD-Conv to perform exceptionally well with respect to small objects and low-resolution images, while its flexibility and efficiency offer strong support for various computer vision applications. In the following sections, we will delve into the principles governing how SPD-Conv operates.

Consider an intermediate feature map X defined by dimensions S × S × C1, where we carry out slicing operations utilizing the SPD structure. This approach systematically retrieves multiple sub-feature maps from various sections of the original feature map X in a sequential manner. The resulting subfeatures are as follows.

$$\begin{gathered} f_{0,0} = X[0:S:scale, 0:S:scale], f_{1,0} = X[1:S:scale, 0:S:scale], \ldots, \\ f_{scale-1,0} = X[scale-1:S:scale, 0:S:scale]; \\ f_{0,1} = X[0:S:scale, 1:S:scale], f_{1,1}, \ldots, \\ f_{scale-1,1} = X[scale-1:S:scale, 1:S:scale]; \\ \ldots\ldots \\ f_{0,scale-1} = X[0:S:scale, scale-1:S:scale], f_{1,scale-1}, \ldots, \\ f_{scale-1,scale-1} = X[scale-1:S:scale, scale-1:S:scale] \end{gathered} \tag{11}$$

When downsampling feature map X, we usually use a scaling factor (scale) to reduce its spatial dimensions. For clarity, as shown in Figure 9, suppose that the scale is set to 2. Feature map X will be divided into four equal sub-maps, respectively, denoted as $X_{0,0}$, $X_{1,0}$, $X_{0,1}$ and $X_{1,1}$, with each sub-map being half the size of the original feature map (i.e., S/2×S/2×c, where S is the spatial size of the original feature map and C is the number of channels). These four sub-maps are then concatenated along the channel dimension to form a new feature map Xˆ. At this point, the number of channels in Xˆ is four times that of the original feature map X, but its spatial dimensions have been halved, resulting in a size of S/2×S/2×4c.

When the feature map X is downsampled, a scaling factor is utilized to reduce its spatial dimensions. In this case, we set the scaling factor to 2, which leads to the feature map X being divided into four smaller sub-maps labeled as $X_{0,0}$, $X_{1,0}$, $X_{0,1}$, and $X_{1,1}$. Each of these sub-maps has spatial dimensions that are half those of the original feature map X; specifically, their sizes are S/2×S/2×c, where S indicates the spatial dimensions of the original feature map and C denotes the number of channels. Following this step,

these four sub-maps are concatenated along the channel axis to create a new feature map referred to as $\hat{X}$. This newly generated feature map $\hat{X}$ has four times as many channels as the original feature map X—specifically, it has 4C channels—while its spatial dimensions have been reduced to S/2×S/2. As a result, the final representation for feature map $\hat{X}$ measures S/2×S/2×4c in size. To summarize, defining a scaling factor and performing division and concatenation operations effectively reduce dimensionality while maintaining information richness.

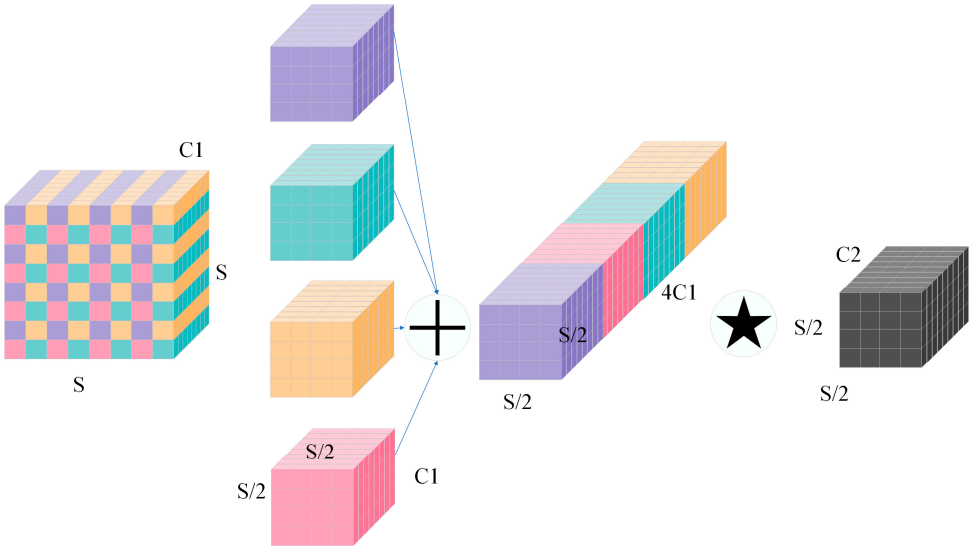


Figure 9. When scale is equal to 2, the feature map is downsampled using the SPD-Conv process.

For different scale values, the dimensions of the feature map after downsampling are defined in accordance with the following equation: S/scale × S/scale × C × scale^2, where the increase in channel number (scale^2) is proportional to the quantity of sub-maps created. To adjust the number of channels and potentially improve feature fusion, we introduce a non-stride 1 × 1 convolution layer with an output channel count set to 2C (assuming that 2C < scale^2 × C to lower the computational complexity and mitigate overfitting risks). After this step, the size of the new feature map $\hat{X}$ becomes S/scale × S/scale × 2C, which maintains information richness while effectively reducing channel counts and possibly enhancing feature expressiveness. In this scenario, the 1×1 convolution layer plays a crucial role by preserving key image features while minimizing information loss, thereby providing more efficient and useful inputs for subsequent layers within the network.

4. Experiments

4.1. Datasets

In this study, to enhance the environmental adaptability and robustness of the small-target detection model, we used two datasets to assess the accuracy of the experimental outcomes. The first dataset is the USOD dataset sourced from FFCA-yolo, while the second is DB_Licenta obtained from the RoboFlow website. The subsequent sections provide comprehensive descriptions of these two datasets.

4.1.1. DB_Licenta

This dataset was selected from a diverse set of 4000 aerial images from the RoboFlow website (Figure 10). These images cover a variety of complex backgrounds, including parks, streets, forests, and snow, enabling the model to accurately identify small targets in various scenarios. We split the dataset in an 8:1:1 ratio, with the training set comprising the majority, consisting of 2672 images and 5863 pedestrian target instances used for model

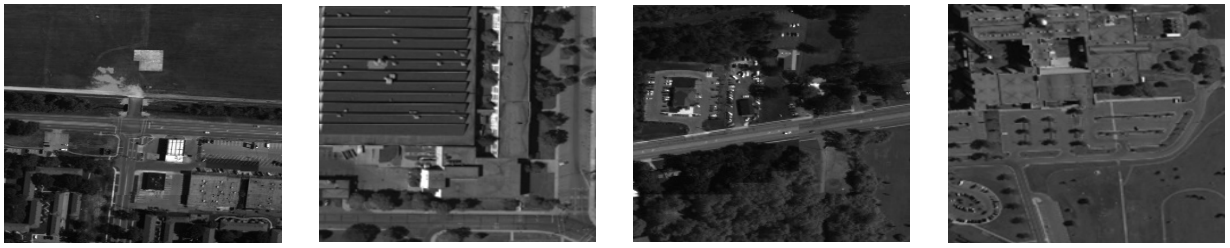
learning and training. The validation set contained 874 images and 1618 instances, which were used to evaluate the model's performance regarding unseen data and adjust model parameters. Finally, the test set consisted of 454 images and 916 instances, which were used to independently validate the final performance of the model to ensure stability and accuracy in real-world applications.

(**a**) (**b**) (**c**) (**d**)

Figure 10. Pedestrian images in the DB_Licenta dataset corresponding to different scenarios: (**a**) park; (**b**) street; (**c**) forest; (**d**) snowfield.

4.1.2. USOD

Another dataset used in this study was the USOD dataset (Figure 11), which is publicly available via the paper on FFCA-YOLO. This dataset contains a total of 3000 images, and it is fully annotated with 43,378 small-target instances, showcasing the diversity of and challenges posed by small targets in remote sensing images. The training, validation, and test sets are split in a ratio of 7:2:1, meaning that the training set consists of 2100 images and their corresponding large quantities of small-target annotations, while the remaining 900 images form the test set, which is used to evaluate the model's generalization ability for unseen data. As shown in Figure 11, one of the most notable features of the USOD dataset is the extreme smallness of its target objects. Specifically, over 96.3% of the target objects in the images have dimensions of less than 16×16 pixels, meaning that these targets are almost impossible to detect in remote sensing images with complex backgrounds. What is more noteworthy is that the proportion of targets with dimensions of less than 32 × 32 pixels is 99.9%, further highlighting the difficulty and importance of detecting these small targets.



(**a**) (**b**) (**c**) (**d**)

Figure 11. Pedestrian images in different scenarios in the USOD database: (**a**) city; (**b**) field; (**c**) town; (**d**) airport.

4.2. Experimental Parameter Configuration

The experiment was conducted using the Windows 10 operating system, PyTorch as the deep learning framework, Python version 3.11, and CUDA version 12.1 to accelerate network speed. Additionally, in the model-training process, the configuration of hyperparameters played a crucial role. The specific configuration is shown in Table 1.

4.3. Evaluation Methodology

In the field of object detection, the choice of evaluation metrics is crucial, as they provide us with the means to quantify the performance of algorithms, allowing us to compare the performance of different models or algorithms for the same dataset. In this

experiment, we used evaluation indicators such as precision, recall, F1-score, and mean average precision (mAP). A summary of these metrics is given below.

Table 1. Hardware and hyperparameter configuration in the model-training process.

Scheme 9.	Configuration Parameters
CPU	Intel Core i9-14900KF
GPU	NVIDIA GeForce RTX 4090, 24GB
Operating System	Windows 10
Epochs	200
Batch Size	16
Optimizer	SGD
Learning Rate	0.01
Decay Coefficient	0.0005
Image Input Size	640 × 640

Precision refers to the proportion of true-positive instances among all samples that a model classifies as positive. This metric is fundamentally associated with a model's ability to correctly identify positive cases, reflecting its proficiency in detecting target objects when it makes a positive prediction. Precision can be calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

In this formula, TP (true positive) refers to the cases in which the model correctly identifies the target object as belonging to the positive class, while FP (false positive) pertains to instances where non-target objects are mistakenly classified as positive. This ratio acts as a direct indicator of a model's accuracy or purity regarding its predictions for positive classes. Generally, high precision suggests that a model demonstrates considerable confidence in its positive classifications; however, it may also correlate with an increased FN (false negative) rate.

Recall rate serves as an essential metric for evaluating a classifier's performance, highlighting the proportion of actual positive samples that are correctly identified by a model. In particular, it is defined as the ratio of true-positive (TP) instances accurately recognized by the classifier to the total number of actual positives (i.e., the TPs plus the false negatives (FNs), which were incorrectly classified). A high recall rate indicates that a classifier can successfully identify a significant number of positive samples; however, this may come at the cost of precision. The formula for calculating the recall rate is shown below:

$$\mathrm{R}ecall = \frac{TP}{TP + \mathrm{FN}} \tag{13}$$

In intricate tasks like multi-class classification and object detection, evaluating model performance comprehensively generally requires the calculation of the average precision (AP) for each class. This process is followed by computing the mean average precision (mAP) through averaging these AP values. The AP is calculated based on the area under the precision–recall curve, reflecting a model's average precision at different recall levels. In contrast, mAP aggregates all category-specific AP scores and provides a unified assessment of model performance across various categories. A higher mAP indicates enhanced overall performance of a model in these categories. Its formula is given below.

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{14}$$

In this paper, two mAP-based evaluation metrics are applied, one of which is mAP50, which represents the mAP at an IoU threshold of 0.5. IoU is a metric that measures the

degree of overlap between the predicted bounding box and the ground-truth bounding box, with a value range of [0,1]. The larger the value, the greater the degree of overlap. The other is mAP50-95, which is a more rigorous evaluation metric that calculates the average mAP within the IoU threshold range from 0.5 to 0.95 (including both ends). This metric can provide a more comprehensive evaluation of a model's performance at different IoU thresholds, reflecting the stability and robustness of a model under different detection difficulties.

The F1-score is a widely used performance evaluation metric in classification tasks that combines the advantages of precision and recall to provide a more comprehensive measure of model performance. Its formula is given below.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{15}$$

The F1-score computes the harmonic mean of precision and recall to equilibrize the performances of both metrics and circumvent the misleading impacts of a solitary metric. Hence, a higher F1-score typically implies that a model is capable of precisely identifying positive samples while minimizing the misclassification of negative samples as positive, indicating an excellent trade-off between the precision and comprehensiveness of a model.

5. Results

In this experiment, we used the two datasets mentioned in Section 4.1 to test our algorithm. We compared our model with several advanced algorithms currently in use in terms of their performances for both datasets. We also conducted an ablation experiment to prove the reliability of the modules we adopted. The experimental process is elaborated upon below.

5.1. Controlled Experiment

5.1.1. Contrastive Experiments on the DB_Licenta Dataset

To assess the performance of the CBGS-YOLO model, we utilized the DB_Licenta dataset and conducted a comparative analysis with state-of-the-art models, including FFCA-YOLO [55], Drone-YOLO [56],UAV-YOLO [57], MS-YOLOv7 [58],GSC-YOLO [59], YOLOv5s [60], YOLOv7-tiny [26], and YOLOv9-s [27]. We examined the box_loss and obj_loss plots (Figure 12a,b) to evaluate the discrepancy between the model's predicted bounding boxes and the ground-truth bounding boxes as well as the predicted object-containing rectangles and the ground-truth boxes. The closer the curves are to zero, the smaller the discrepancy between the predicted and actual values. The loss values of CBGS-YOLO are notably closer to zero, indicating superior training performance. Additionally, Figure 12c–f demonstrate that CBGS-YOLO consistently outperformed the other models in regard to four key metrics, thereby validating its high detection efficiency for this dataset.

The experimental results described in Table 2 show that the CBGS-YOLO model outperformed all the other models in terms of all five evaluation metrics (precision, recall, F1 score, mAP50, and mAP50-95). Compared with the best performance of the baseline model, it led to increases of 4.2%, 1.3%, 2.6%, 1.8%, and 4.1%, respectively, in the five metrics. In terms of detection speed, we used FPS (frames per second) as the evaluation criterion, and through comparative experiments, we found that the indicator of CBGS-YOLO was also significantly better than the other indicators. This was mainly due to the introduction of the c3 structure of the CBAM attention mechanism in the main stem layer, a feature that enhanced the model's ability to identify small targets. The addition of the SPD-Conv structure to the concat module in the neck layer improved the model's detection performance for low-resolution images. The reduction in the number of parameters was

mainly due to replacing traditional convolution layers with Ghost modules. In summary, the CBGS-YOLO model proposed in this paper significantly improves the accuracy of detecting small targets while also maintaining a lightweight design.

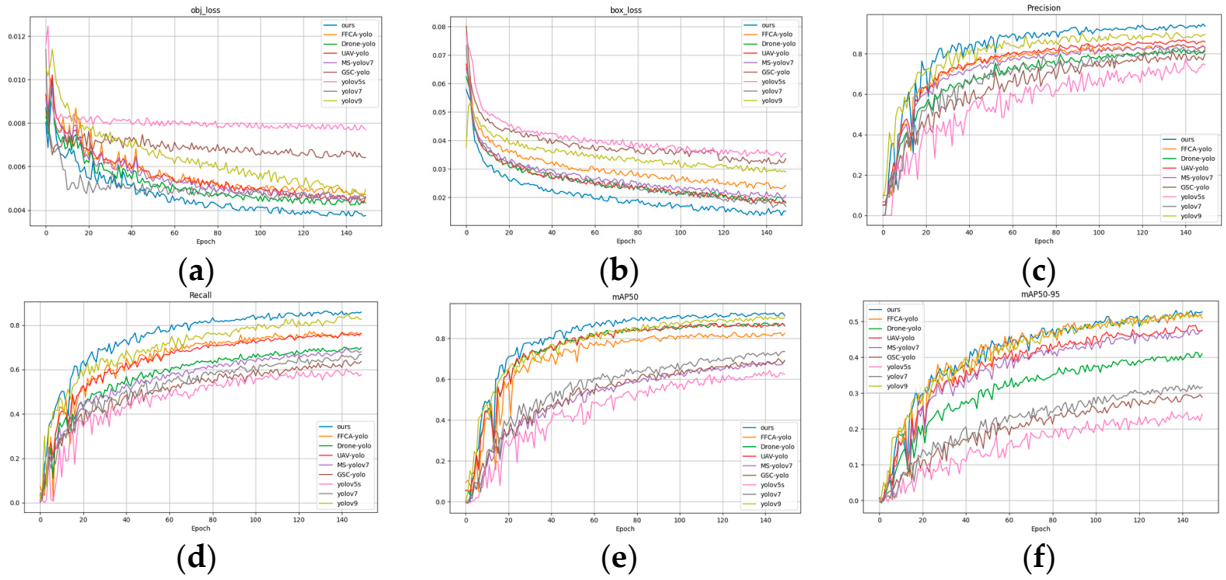


Figure 12. Comparison of training performance metrics between CBGS-YOLO and FFCA-YOLO, Drone-YOLO, UAV-YOLO, MS-YOLOv7, GSC-YOLO, yolov5s, yolov7-tiny, and yolov9-s on the DB_Licenta dataset. (**a**) obj_loss. (**b**) box_loss. (**c**) Precision. (**d**) Recall. (**e**) mAP50. (**f**) mAP50-95.

Table 2. A comparison of the other eight models and CBGS-YOLO on the test data of the DB_Licenta dataset.

Method	Precision	Recall	F1	mAP50	mAP50-95	Parameters	FPS
FFCA-YOLO	0.847	0.771	0.807	0.825	0.395	6.31	657.420
Drone-YOLO	0.805	0.682	0.738	0.874	0.402	6.52	685.910
UAV-YOLO	0.864	0.754	0.805	0.873	0.484	6.99	623.861
MS-YOLOv7	0.837	0.675	0.747	0.674	0.457	6.03	586.606
GSC-YOLO	0.873	0.642	0.740	0.693	0.296	6.09	639.963
Yolov5s	0.773	0.603	0.677	0.638	0.241	7.02	**738.010**
Yolov7-tiny	0.824	0.637	0.719	0.731	0.323	8.62	510.881
Yolov9-s	0.905	0.849	0.876	0.904	0.472	7.10	325.610
ours	**0.947**	**0.862**	**0.902**	**0.921**	**0.525**	**5.12**	690.908

Figure 13 offers an exhaustive comparison of the small-target detection capabilities of six models, including CBGS-YOLO, in complex and variable scenarios. The scenes from left to right are as follows: a highly intricate street backdrop, a forest environment densely populated with trees, a complex scene featuring overlapping multi-task targets, and a low-pixel challenge. Through analysis, it was determined that the contrasts in the first three scenes were particularly pronounced: While other models have faced problems of varying degrees in terms of false detection and missed detection, the CBGS-YOLO model manifested superior detection ability without any instances of false or failed detection. This finding thoroughly substantiates the superiority and robustness of our model with regard to complex environments. In the fourth low-pixel target detection group, all the models involved in the comparison successfully identified the targets. Notably, the CBGS-YOLO model achieved not only high accuracy but also a detection confidence of the highest level: 0.93. This outcome undoubtedly highlights the exceptional detection performance of our model under this condition, further validating its superior and effective performance in handling tasks related to small-target detection.

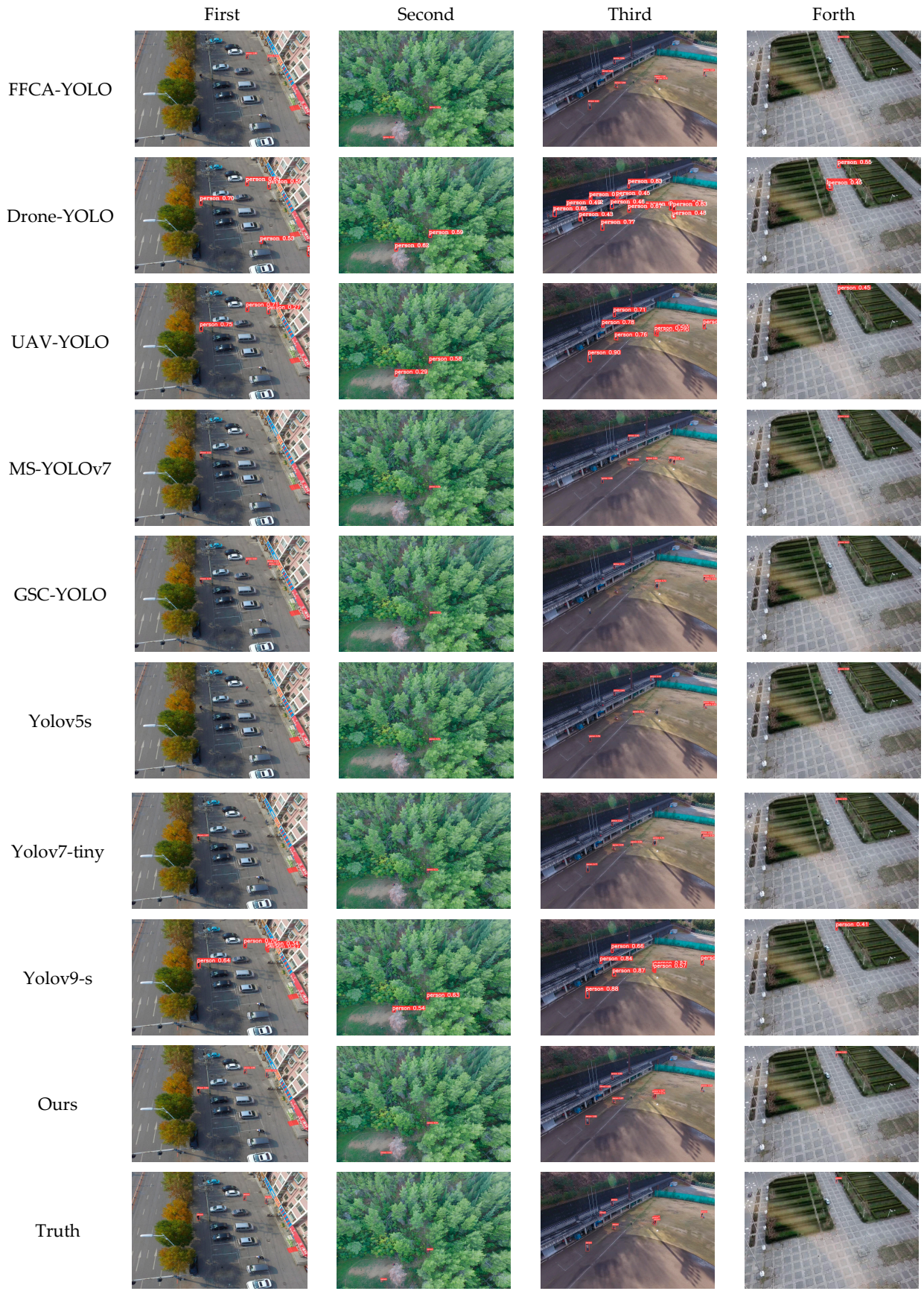


Figure 13. This chart provides a comparative analysis of the CBGS-YOLO model against other models, demonstrating its superior performance in minimizing false negatives and false positives while achieving high-confidence detections.

5.1.2. Contrastive Experiments on the USOD Dataset

To further assess the effectiveness and robustness of the CBGS-YOLO model, we selected a new dataset, USOD, with distinct characteristics compared to the DB_Licenta dataset for an in-depth analysis. A notable feature of this new dataset is its lower image resolution and grayscale presentation, contrasting with the color images found in the DB_Licenta dataset. This alteration introduces additional challenges to the detection task. By conducting systematic evaluations of the latest YOLO variants, including FFCA-YOLO, GSC-YOLO, Drone-YOLO, and classic YOLOv5s, using a new dataset, we can objectively demonstrate the superior detection ability and optimization effects of the CBGS-YOLO model in complex and low-quality image environments. This study not only validates the generality of CBGS-YOLO, but also highlights its significant advantage in handling challenging grayscale and low-resolution images.

Figure 14a,b provide a visual representation of the trends of box_loss and obj_loss, where the loss curve of the CBGS-YOLO model is closer to zero, which is a significant feature that directly reflects the high efficiency of the training process. Further observation of the performance curves in Figure 14c–f reveals that over time, CBGS-YOLO consistently demonstrates significant advantages over the other eight models for all four evaluation metrics. This result not only strongly proves the CBGS-YOLO model's powerful detection ability for complex datasets, but also highlights its significant achievements in improving detection accuracy and overall performance.

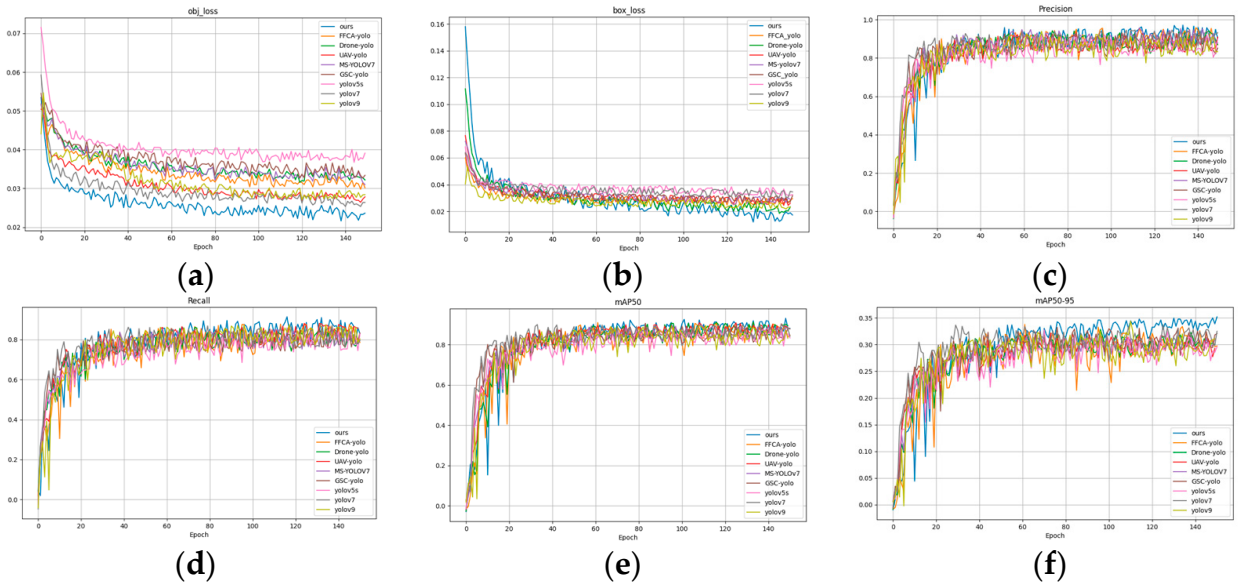


Figure 14. Comparison of training performance metrics between CBGS-YOLO and FFCA-YOLO, Drone-YOLO, UAV-YOLO, MS-YOLOv7, GSC-YOLO, yolov5s, yolov7-tiny, and yolov9-s on the USOD dataset. (**a**) obj_loss. (**b**) box_loss. (**c**) Precision. (**d**) Recall. (**e**) mAP50. (**f**) mAP50-95.

The experimental results presented in Table 3 clearly demonstrate that the CBGS-YOLO model outperformed all the other models across all five evaluation metrics, with improvements of 2.7%, 4.5%, 3.6%, 1.5%, and 1.1% compared to the best-performing model in the control group. These findings strongly validate this model's ability to precisely detect small targets in grayscale images. In terms of detection speed and model size, the CBGS-YOLO model also exhibited superior performance. In summary, the CBGS-YOLO model proposed in this paper significantly enhanced the accuracy of detecting small targets while also maintaining a lightweight design and efficient detection capabilities.

Table 3. A comparison of the other eight models and CBGS-YOLO on the test data of the USOD dataset.

Method	Precision	Recall	F1	mAP50	mAP50-95	Parameters	FPS
FFCA-YOLO	0.921	0.888	0.904	0.890	0.312	6.31	344.336
Drone-YOLO	0.897	0.839	0.867	0.901	0.302	6.32	398.597
UAV-YOLO	0.841	0.797	0.818	0.893	0.277	5.99	409.896
MS-YOLOv7	0.863	0.858	0.860	0.885	0.298	6.22	443.376
GSC-YOLO	0.891	0.815	0.851	0.873	0.331	6.09	396.420
Yolov5s	0.881	0.791	0.834	0.853	0.317	7.02	**472.543**
Yolov7-tiny	0.907	0.826	0.865	0.884	0.336	8.62	399.951
Yolov9-s	0.893	0.831	0.861	0.896	0.334	7.10	255.791
ours	**0.948**	**0.883**	**0.914**	**0.916**	**0.347**	**5.12**	456.077

Figure 15 provides a comprehensive comparison of the small-target detection capabilities of nine models, including CBGS-YOLO, across four complex scenarios: multiple-target dispersion, moving targets, occluded targets, and multiple-target stacking. Analysis reveals that the other models had false or failed detections in various scenarios, whereas CBGS-YOLO demonstrated a superior performance, with no false or failed detections and the highest detection confidence. This evidence underscores the excellent detection performance and robustness of CBGS-YOLO in complex environments. Not only does it achieve high detection accuracy, but its confidence is also high, further validating this model's advanced and effective capacity for small-target detection.

5.2. Ablation Experiment

To comprehensively and meticulously verify the effectiveness of each improvement strategy in regard to the target detection task, we elaborately devised a series of ablation experiments based on the baseline model (YOLOv5-s). In these ablation experiments, we successively introduced three advanced enhancement technologies: the Convolutional Block Attention Model (CBAM), the Ghost module, and Spatial Pyramid Depthwise Separable Convolution (SPDConv). Each technology was introduced to enhance the model's performance with regard to various dimensions, such as strengthening its feature extraction ability, reducing computational complexity, and optimizing the use of spatial information. Specifically, CBAM integrates the channel attention and spatial attention mechanisms, enabling the model to focus more on the key regions in an image, thereby enhancing detection accuracy. The Ghost module efficiently utilizes feature maps to achieve an effect similar to traditional convolution but with a lower computational cost, contributing to improving the model's inference speed. Meanwhile, SPDConv constructs a spatial pyramid structure, allowing the model to capture multi-scale spatial information and further strengthening its adaptability to complex scenes.

To quantitatively assess the influence of these modules on the model's performance, we selected several crucial performance indicators, including the precision, recall, F1 score, mean average precision (mAP), and model parameters (parameters). These indicators comprehensively reflected the model's performance in the detection task. The dataset employed in this experiment was USOD, and by conducting ablation experiments on this dataset, we were able to more accurately evaluate the effects and limitations of each improvement strategy in practical applications, providing substantial support for subsequent research and optimization.

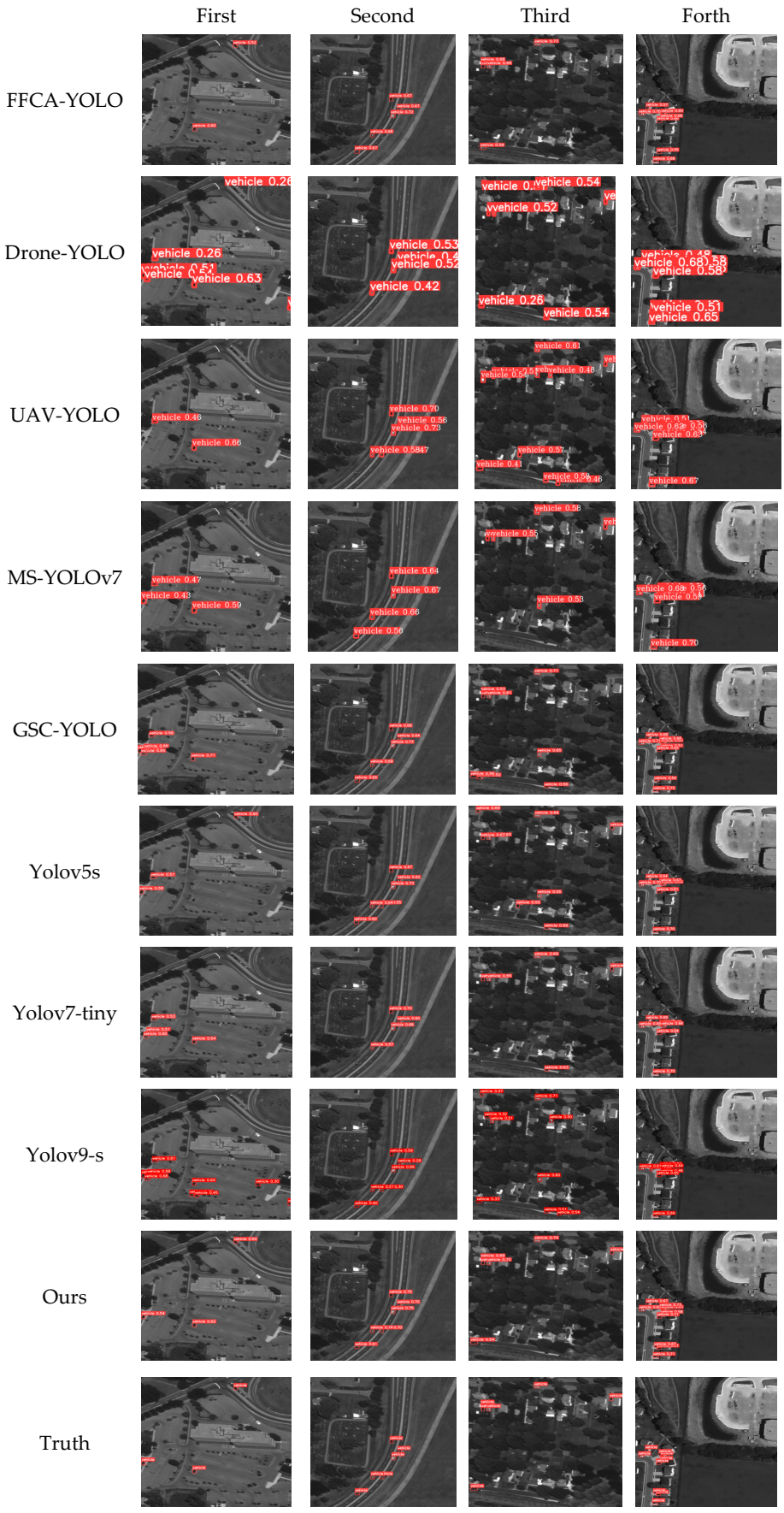


Figure 15. In four scenarios, CBGS-YOLO outperformed other models in minimizing errors and achieving high confidence.

5.2.1. Ghost

Table 4 presents the detailed ablation experiment outcomes regarding the Ghost module within the YOLOv5 model. By substituting the standard convolution block (CBS) in YOLOv5 with the Ghost_CBS module, we managed to reduce the model parameters by approximately 47%, significantly lowering the model's complexity. Nevertheless, this parameter optimization strategy also entailed a marginal sacrifice in detection performance. Specifically, the precision dropped from 0.881 to 0.826, the recall declined from 0.791 to 0.664, and the mAP and F1 scores decreased to 0.802 and 0.736, respectively. Despite the minor performance deterioration, this alteration still lies within an acceptable range when compared to the potential for improvement.

Table 4. Performance comparison of lightweight operations after networking.

Method	Precision	Recall	mAP50	F1-Score	Parameters
YOLOV5s	0.881	0.791	0.853	0.834	7.02M
YOLOV5s+Ghost_CBS	0.826	0.664	0.802	0.736	3.68M
YOLOV5s+AK_CBS	0.625	0.423	0.665	0.505	3.52M

To further validate the advantage of the Ghost module in terms of reducing the number of parameters while maintaining performance, we carried out a comparative experiment by replacing the Ghost_CBS module with another convolution module, AKConv [61] (i.e., AK_CBS). The experimental results indicate that, although AK_CBS also reduced the model parameters to a certain extent, it led to a more pronounced performance decline. Specifically, the precision plummeted sharply to 0.625, and the recall was merely 0.423, while the mAP and F1 scores also dropped to 0.665 and 0.505, respectively. This comparison strikingly highlights the superiority of the Ghost module in balancing model complexity and detection performance.

In conclusion, the Ghost module not only effectively reduces the number of parameters in the YOLOv5 model, but also, in comparison with other convolution replacement schemes, results in a relatively smaller decline in detection performance, providing a firmer foundation for subsequent model optimization and improvement.

5.2.2. SPDConv

To verify the effectiveness of the SPD-Conv module, we developed the model introduced in Section 5.2.1 and added the SPD module to the model in two steps: the first step involved adding it only to the trunk, and the second step involved adding it to both the backbone and the neck. The experimental results are shown in Table 5. The effectiveness of both methods was validated. When the SPD module was only added to the backbone, the model's accuracy improved from 0.826 to 0.893, its recall rate improved from 0.664 to 0.852, and its mAP improved from 0.802 to 0.847. When the SPD module was added to both the backbone and the neck, compared with adding only the SPD module to the trunk, the model's accuracy improved from 0.893 to 0.901, its recall rate improved from 0.852 to 0.879, and its mAP improved from 0.847 to 0.884.

Table 5. The effectiveness of SPD modules can be verified by adding them to different locations within the network.

Method	Precision	Recall	mAP50	F1-Score	Parameters
YOLOV5s+Ghost_CBS	0.826	0.664	0.802	0.736	3.68M
Backbone+SPDConv	0.893	0.852	0.847	0.872	4.52M
Backbone, Neck+SPDConv	0.901	0.879	0.884	0.889	4.81M

The experimental results strongly prove that the SPD module can effectively enhance this model's ability to detect small targets. Since our goal was to increase the effectiveness of the network's target detection ability, we chose to further study the model by adding the SPD module to the backbone and the neck.

5.2.3. Convolutional Block Attention Model (CBAM)

In this study, we thoroughly investigated the efficacy of the convolutional block attention module (CBAM) in augmenting model performance and endeavored to comprehensively substantiate the superiority of CBAM over other prevalent attention mechanisms by means of a sequence of comparative experiments. We meticulously chose four representative attention mechanisms as benchmarks, namely, the Global Attention Module (GAM) [62,63], Coordinate Attention (CA) [63], Efficient Channel Attention (ECA) [64], and Hybrid Local Channel Attention (MLCA) [65], in order to establish a comprehensive and impartial evaluation framework.

The experimental outcomes (presented in Table 6) reveal that not all the introduced attention mechanisms could positively enhance the performance of this model. Specifically, we discerned that the incorporation of Efficient Channel Attention (ECA) actually led to a marginal reduction in the model's mean average precision (mAP), dropping from 0.901 to 0.896, signifying that the selection and configuration of attention mechanisms can exert a crucial influence on model performance in certain circumstances. Likewise, the integration of hybrid local channel attention (MLCA) did not exert a substantial impact on the model's mAP, a result that might reflect the incomplete compatibility between the mechanism and the current model architecture or task attributes. In contrast, the CBAM module manifested notable superiority. Under identical experimental conditions, the introduction of CBAM elevated the model's mAP to 0.916, constituting the most prominent performance gain. This finding validates CBAM's potent ability to capture crucial feature information in both the spatial and channel dimensions.

Table 6. By adding different attention mechanism modules to the network to verify its effectiveness.

Method	Precision	Recall	mAP50	F1-Score	Parameters
Backbone, Neck+SPDConv	0.901	0.879	0.884	0.889	4.81 M
+GAM	0.927	0.862	0.903	0.893	5.07 M
+CA	0.914	0.855	0.908	0.884	5.19 M
+ECA	0.896	0.861	0.902	0.878	5.04 M
+MLCA	0.902	0.877	0.911	0.894	**5.23 M**
+CBAM (ours)	**0.938**	**0.883**	**0.916**	**0.909**	5.12 M

5.2.4. Comparison of YOLO Models

To validate the rationale behind selecting YOLOv5 over the more advanced YOLOv10 as a submodel, we conducted a comparative experiment (Table 7). The results indicate that the performances of YOLOv5 and YOLOv10 are comparable with regard to detecting objects in the USOD dataset. However, the parameters metric of YOLOv10 was notably higher than that of YOLOv5, contradicting the initial design objectives of our model. Consequently, YOLOv5 was deemed more appropriate for this model design.

Table 7. A comparative experiment between YOLOV5 and YOLOV10 as submodels.

Method	Precision	Recall	mAP50	F1-Score	Parameters
CBGS-YOLOV5	0.948	**0.883**	**0.916**	0.914	5.12 M
CBGS-YOLOV10	**0.956**	0.882	0.913	**0.917**	**9.03 M**

5.3. Discussion

In the field of detecting small targets in remote sensing images, a series of breakthrough algorithms have emerged due to the continuous optimization of technology, greatly promoting progress in this field. This paper focuses on this hotspot and deeply analyzes the excellent performance of the CBGS-YOLO model in color remote sensing image target detection through carefully designed comparative experiments. In the experiment, five remote sensing image small-target detection models and three general target detection models were selected as comparison benchmarks, and a comprehensive evaluation was carried out with regard to the DB_Licenta dataset. The results show that the CBGS-YOLO model showed high precision and a high recall rate in various types of target detection, especially in complex scenes, such as city blocks and mountain forests. Regardless of whether the target was blocked, the model showed excellent detection performance, fully verifying its wide applicability and robustness in detecting targets in color remote sensing images.

In order to further explore this model's detection ability with regard to gray-scale images, we used the USOD dataset for supplementary experiments. The experimental data obtained show that the CBGS-YOLO model also performed well when applied to gray-scale images and could accurately complete small-target detection tasks in the face of complex situations such as multi-target dispersion, moving targets, blocked targets, and multi-target stacking, further proving its strong robustness and generalization ability.

Then, in order to verify the reliability of each module selected in the model, we conducted a series of ablation experiments and four groups of relevant comparative tests. First, in the ablation experiment related to Ghost, we conducted two groups of relevant experiments. The test results show that the Ghost module effectively reduced about 47% of the model parameters for YOLOv5. At the same time, the corresponding detection performance was relatively good and only slightly decreased. A comparison experiment involving the AKConv module showed that the Ghost module could reduce the model's complexity and better maintain the detection performance index, reflecting its superiority. In the ablation experiment conducted using SPDConv, we also conducted two groups of ablation experiments, and the experimental results showed that the SPD-Conv module significantly improved the model's performance, especially after it was added to the trunk and neck at the same time; the model's accuracy, recall rate, and mAP were significantly improved, effectively enhancing the model's ability to recognize small objects. In the ablation experiment on attention mechanisms, we comprehensively evaluated the performance of CBAM and the other four attention mechanisms through comparative experiments and found that not all the attention mechanisms could improve the model's performance, so caution should be exercised when selecting them. CBAM showed significant advantages, effectively increasing the model's mAP to 0.916, verifying its powerful ability to capture important feature information. Finally, in the experiment regarding why YOLOv5 should be used instead of YOLOv10, both exhibited similar performances with regard to detecting targets in the USOD dataset, but YOLOv5 had fewer parameters, which is more in line with the original design intention of the model. Therefore, it is more reasonable to choose YOLOv5 as the sub-model.

To sum up, the CBGS-YOLO model shows comprehensive advantages in the field of small-target detection for remote sensing images and can quickly and accurately complete detection tasks relating to both color and gray images. Its excellent performance, wide applicability, and strong generalization ability grant it broad application prospects and great practical value in this field.

6. Conclusions

In this paper, we put forward an innovative model of a lightweight network for detecting objects in remote sensing images, which is distinguished by our substitution of the traditional CBS module with the Ghost_CBS module. The core concept of this model lies in reducing the quantity of model parameters without compromising the ability to detect small targets, thereby allowing for more efficient computations and deployment. Moreover, by deeply integrating the CBAM attention mechanism with the C3 module, we gave our model the ability to intelligently assess the significance of each candidate box or region and prioritize focusing on the regions that are most likely to contain target information, thereby conspicuously enhancing the precision and efficiency of detection. Additionally, this model incorporates the SPDConv module into its backbone and neck, effectively enhancing its adaptability to low-resolution images and detection performance, enabling excellent detection outcomes to be maintained even under circumstances of limited image quality. The experimental results indicate that, in terms of color image detection, the proposed model surpasses the next-best model in terms of precision, recall, F1, mAP@50, and mAP@50-95 by 0.042, 0.013, 0.026, 0.018, and 0.041, respectively. In grayscale image detection, it also outperforms the next-best model in these metrics by 0.017, 0.025, 0.042, 0.015, and 0.011. The ablation experiment further validates the efficacy of the proposed enhancements, not only boosting the model's detection performance but also maintaining its lightweight feature. The proposed model not only exhibits excellent accuracy but also realizes lightweighting, making it more suitable for applications in embedded systems with limited resources.

Looking ahead, our research will focus on two main directions: The first is to comprehensively explore the stable detection strategies for moving targets in video images, overcoming the limitations of single-frame detection in video applications. The second is to study how to utilize the temporal information in sequence images to optimize the detection process; reduce the false-alarm rate, especially in relation to complex scenes and low-quality images; and further enhance the model's ability to detect weak and small targets.

Author Contributions: Conceptualization, methodology, software, validation, Z.W.; formal analysis, investigation, resources, N.L.; writing—original draft preparation, J.Y.; writing—review and editing, D.W.; supervision, X.Y.; project administration, Y.G.; funding acquisition, W.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Jilin Province Youth Growth Science and Technology Program Project (No. 20220508041RC).

Data Availability Statement: The data used in this study are available upon request from the corresponding author via email.

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

References

1. Zhang, J.; Zhang, X.; Tan, X.; Yuan, X. Extraction of urban built-up area based on deep learning and multi-sources data fusion—The application of an emerging technology in urban planning. *Land* **2022**, *11*, 1212. [CrossRef]
2. Chaturvedi, V.; de Vries, W.T. Machine learning algorithms for urban land use planning: A review. *Urban Sci.* **2021**, *5*, 68. [CrossRef]
3. Patil, P. Applications of deep learning in traffic management: A review. *Int. J. Bus. Intell. Big Data Anal.* **2022**, *5*, 16–23.
4. Li, H.; Yu, L.; Zhang, J.; Lyu, M. Fusion deep learning and machine learning for heterogeneous military entity recognition. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1103022. [CrossRef]
5. Mohamed, D.A. Comparative Study of Deep Learning Models for Unimodal\& Multimodal Disaster Data for Effective Disaster Management. Ph.D. Thesis, The British University in Dubai (BUiD), Dubai, United Arab Emirates, 2021.

6. Han, J.; Ma, Y.; Zhou, B.; Fan, F.; Liang, K.; Fang, Y. A robust infrared small target detection algorithm based on human visual system. *IEEE Geosci. Remote Sens. Lett.* **2014**, *11*, 2168–2172.
7. Luo, X.; Wu, Y.; Zhao, L. YOLOD: A target detection method for UAV aerial imagery. *Remote Sens.* **2022**, *14*, 3240. [CrossRef]
8. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
9. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–26 June 2005; pp. 886–893.
10. Papageorgiou, C.P.; Oren, M.; Poggio, T. A general framework for object detection. In Proceedings of the Sixth Internation-al Conference on Computer Vision, Bombay, India, 7 January 1998; pp. 555–562.
11. Wang, X.; Wu, S.; Liu, Y. Detecting wood surface defects with fusion algorithm of visual saliency and local threshold segmentation. In *Ninth International Conference on Graphic and Image Processing*; SPIE: Bellingham, WA, USA, 2018; pp. 529–537.
12. Mittal, M.; Verma, A.; Kaur, I.; Kaur, B.; Sharma, M.; Goyal, L.M.; Roy, S.; Kim, T.-H. An efficient edge detection approach to provide better edge connectivity for image analysis. *IEEE Access* **2019**, *7*, 33240–33255. [CrossRef]
13. Brown, M.; Szeliski, R.; Winder, S. Multi-image matching using multi-scale oriented patches. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; pp. 510–517.
14. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
15. 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]
16. Wang, H.; Li, Z.; Ji, X.; Wang, Y. Face r-cnn. *arXiv* **2017**, arXiv:1706.01061.
17. 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.* **2016**, *39*, 1137–1149. [CrossRef]
18. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
19. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
20. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
21. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
22. Farhadi, A.; Redmon, J. *Yolov3: An Incremental Improvement*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 1–6.
23. Bochkovskiy, A.; Wang, C.; Liao, H.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
24. Mokayed, H.; Nayebiastaneh, A.; De, K.; Sozos, S.; Hagner, O.; Backe, B. Nordic Vehicle Dataset (NVD): Performance of vehicle detectors using newly captured NVD from UAV in different snowy weather conditions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Vancouver, BC, Canada, 17–24 June 2023; pp. 5314–5322.
25. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A single-stage object detection framework for industrial applications. *arXiv* **2022**, arXiv:2209.02976.
26. Wang, C.; Bochkovskiy, A.; Liao, H.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
27. Wang, C.; Yeh, I.; Liao, H.M. *Yolov9: Learning What YOU Want to learn Using Programmable Gradient Information*; Springer: Berlin/Heidelberg, Germany, 2025; pp. 1–21.
28. Wang, A.; Chen, H.; Liu, L.; Chen, K.; Lin, Z.; Han, J.; Ding, G. Yolov10: Real-time end-to-end object detection. *arXiv* **2024**, arXiv:2405.14458.
29. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. *Ssd: Single Shot Multibox Detector*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
30. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional single shot detector. *arXiv* **2017**, arXiv:1701.06659.
31. Li, Z.; Yang, L.; Zhou, F. FSSD: Feature fusion single shot multibox detector. *arXiv* **2017**, arXiv:1712.00960.
32. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 10–17 October 2021; pp. 10012–10022.
33. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. *End-to-End Object Detection with Transformers*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 213–229.

34. Rozantsev, A.; Lepetit, V.; Fua, P. Flying objects detection from a single moving camera. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4128–4136.
35. Sun, H.; Sun, X.; Wang, H.; Li, Y.; Li, X. Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model. *IEEE Geosci. Remote Sens. Lett.* **2011**, *9*, 109–113. [CrossRef]
36. Deng, H.; Sun, X.; Liu, M.; Ye, C.; Zhou, X. Small infrared target detection based on weighted local difference measure. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 4204–4214. [CrossRef]
37. Zhang, L.; Zhang, L.; Tao, D.; Huang, X.; Du, B. Hyperspectral remote sensing image subpixel target detection based on supervised metric learning. *IEEE Trans. Geosci. Remote Sens.* **2013**, *52*, 4955–4965. [CrossRef]
38. Tan, L.; Lv, X.; Lian, X.; Wang, G. YOLOv4_Drone: UAV image target detection based on an improved YOLOv4 algorithm. *Comput.\Electr. Eng.* **2021**, *93*, 107261. [CrossRef]
39. Xu, J.; Tan, X.; Luo, R.; Song, K.; Li, J.; Qin, T.; Liu, T.Y. NAS-BERT: Task-agnostic and adaptive-size BERT compression with neural architecture search. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual, 14–18 August 2021; pp. 1933–1943.
40. Liu, Y.; Zhang, W.; Wang, J. Zero-shot adversarial quantization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 1512–1521.
41. Zhu, J.; Tang, S.; Chen, D.; Yu, S.; Liu, Y.; Rong, M.; Yang, A.; Wang, X. Complementary relation contrastive distillation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9260–9269.
42. Wimmer, P.; Mehnert, J.; Condurache, A. Interspace pruning: Using adaptive filter representations to improve training of sparse cnns. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 12527–12537.
43. Hanson, S.; Pratt, L. Comparing biases for minimal network construction with back-propagation. *Adv. Neural Inf. Process. Syst.* **1988**, *1*.
44. Courbariaux, M.; Bengio, Y.; David, J.P. Binaryconnect: Training deep neural networks with binary weights during propagations. *Adv. Neural Inf. Process. Syst.* **2015**, *28*.
45. Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or−1. *arXiv* **2016**, arXiv:1602.02830.
46. Jaderberg, M.; Vedaldi, A.; Zisserman, A. Speeding up convolutional neural networks with low rank expansions. *arXiv* **2014**, arXiv:1405.3866.
47. Shen, C.; Xue, M.; Wang, X.; Song, J.; Sun, L.; Song, M. Customizing student networks from heterogeneous teachers via adaptive knowledge amalgamation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3504–3513.
48. Howard, A.G. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
49. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
50. Wu, B.; Wan, A.; Yue, X.; Jin, P.; Zhao, S.; Golmant, N.; Gholaminejad, A.; Gonzalez, J.; Keutzer, K. Shift: A zero flop, zero parameter alternative to spatial convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9127–9135.
51. Jeon, Y.; Kim, J. Constructing fast network through deconstruction of convolution. *Adv. Neural Inf. Process. Syst.* **2018**, *31*.
52. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
53. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. Ghostnet: More features from cheap operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1580–1589.
54. Yang, Z.; Wu, Q.; Zhang, F.; Zhang, X.; Chen, X.; Gao, Y. A New Semantic Segmentation Method for Remote Sensing Images Integrating Coordinate Attention and SPD-Conv. *Symmetry* **2023**, *15*, 1037. [CrossRef]
55. Zhang, Y.; Ye, M.; Zhu, G.; Liu, Y.; Guo, P.; Yan, J. FFCA-YOLO for small object detection in remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 1–15. [CrossRef]
56. Zhang, Z. Drone-YOLO: An efficient neural network method for target detection in drone images. *Drones* **2023**, *7*, 526. [CrossRef]
57. Liu, M.; Wang, X.; Zhou, A.; Fu, X.; Ma, Y.; Piao, C. Uav-yolo: Small object detection on unmanned aerial vehicle perspective. *Sensors* **2020**, *20*, 2238. [CrossRef]
58. Doherty, J.; Gardiner, B.; Kerr, E.; Siddique, N. MS-YOLOv7: One-Stage Object Detection Integrating Bi-Directional Feature Pyramid Networks. *Pattern Recognit.* **2024**, *160*, 111209. [CrossRef]
59. Liu, S.; Cao, L.; Li, Y. Lightweight pedestrian detection network for UAV remote sensing images based on strideless pooling. *Remote Sens.* **2024**, *16*, 2331. [CrossRef]

60. Jocher, G.; Chaurasia, A.; Stoken, A.; Stoken, A.; Borovec, J.; Kwon, Y.; Michael, K.; Fang, J.; Wong, C.; Yifu, Z.; et al. ultralytics/yolov5: V6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations. *Zenodo* **2022**.
61. Zhang, X.; Song, Y.; Song, T.; Yang, D.; Ye, Y.; Zhou, J.; Zhang, L. AKConv: Convolutional kernel with arbitrary sampled shapes and arbitrary number of parameters. *arXiv* **2023**, arXiv:2311.11587.
62. Liu, Y.; Shao, Z.; Hoffmann, N. Global attention mechanism: Retain information to enhance channel-spatial interactions. *arXiv* **2021**, arXiv:2112.05561.
63. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722.
64. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11534–11542.
65. Wan, D.; Lu, R.; Shen, S.; Xu, T.; Lang, X.; Ren, Z. Mixed local channel attention for object detection. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106442. [CrossRef]