


Article

# YOLO-DroneMS: Multi-Scale Object Detection Network for Unmanned Aerial Vehicle (UAV) Images

Xueqiang Zhao <sup>1,2</sup> and Yangbo Chen <sup>1,\*</sup> 

<sup>1</sup> School of Geography and Planning, Sun Yat-sen University, Guangzhou 510275, China; zhaoxq28@mail2.sysu.edu.cn

<sup>2</sup> China Water Resources Pearl River Planning Surveying & Designing Co., Ltd., Guangzhou 510610, China

\* Correspondence: eescyb@mail.sysu.edu.cn

**Abstract:** In recent years, research on Unmanned Aerial Vehicles (UAVs) has developed rapidly. Compared to traditional remote-sensing images, UAV images exhibit complex backgrounds, high resolution, and large differences in object scales. Therefore, UAV object detection is an essential yet challenging task. This paper proposes a multi-scale object detection network, namely YOLO-DroneMS (You Only Look Once for Drone Multi-Scale Object), for UAV images. Targeting the pivotal connection between the backbone and neck, the Large Separable Kernel Attention (LSKA) mechanism is adopted with the Spatial Pyramid Pooling Factor (SPPF), where weighted processing of multi-scale feature maps is performed to focus more on features. And Attentional Scale Sequence Fusion DySample (ASF-DySample) is introduced to perform attention scale sequence fusion and dynamic upsampling to conserve resources. Then, the faster cross-stage partial network bottleneck with two convolutions (named C2f) in the backbone is optimized using the Inverted Residual Mobile Block and Dilated Reparam Block (iRMB-DRB), which balances the advantages of dynamic global modeling and static local information fusion. This optimization effectively increases the model's receptive field, enhancing its capability for downstream tasks. By replacing the original Ciou with WIoUv3, the model prioritizes anchoring boxes of superior quality, dynamically adjusting weights to enhance detection performance for small objects. Experimental findings on the VisDrone2019 dataset demonstrate that at an Intersection over Union (IoU) of 0.5, YOLO-DroneMS achieves a 3.6% increase in mAP@50 compared to the YOLOv8n model. Moreover, YOLO-DroneMS exhibits improved detection speed, increasing the number of frames per second (FPS) from 78.7 to 83.3. The enhanced model supports diverse target scales and achieves high recognition rates, making it well-suited for drone-based object detection tasks, particularly in scenarios involving multiple object clusters.

**Keywords:** drone images; LSKA; DySample; iRMB-DRB; WIoU



**Citation:** Zhao, X.; Chen, Y. YOLO-DroneMS: Multi-Scale Object Detection Network for Unmanned Aerial Vehicle (UAV) Images. *Drones* **2024**, *8*, 609. <https://doi.org/10.3390/drones8110609>

Academic Editor: Seokwon Yeom

Received: 29 August 2024

Revised: 8 October 2024

Accepted: 23 October 2024

Published: 24 October 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The integration of Unmanned Aerial Vehicles (UAVs) and industry applications is gradually becoming a research trend. Achieving accurate and real-time detection and tracking of UAV multi-targets is one of the core challenges widely encountered in fields such as security patrol, agricultural pest control, IoT transportation, and power line inspection [1]. Unlike traditional object detection, UAV imagery often contains numerous small objects that are densely distributed, exhibits significant scale variations, and is set against complex backgrounds. These factors significantly hinder the precision of detection models. Additionally, the high-resolution characteristics of UAV images complicate the optimization of object detection algorithms. Some methods struggle to effectively detect and differentiate small objects due to the cluttered environments and varying scales, leading to a higher rate of false positives and negatives. Furthermore, many current approaches do not

fully exploit the spatial resolution and contextual information available in high-resolution images, resulting in suboptimal performance in object detection tasks [2,3].

With the rapid development of deep-learning methods, deep-learning-based object detection has far surpassed the performance of traditional methods. Deep-learning-based generic object detection algorithms can be categorized into two types: the R-CNN (Region-based Convolutional Neural Network) series two-stage algorithms and the YOLO (You Only Look Once) SSD (Single-Shot MultiBox Detector) series one-stage algorithms [4–6]. One-stage detectors provide end-to-end performance advantages; however, they typically demonstrate lower accuracy in localizing and recognizing small objects. In contrast, two-stage object detectors, which utilize a locate-then-recognize framework, achieve higher accuracy but suffer from inferior real-time performance. Given the real-time operational demands of UAVs, it is crucial to tackle the challenge of lightweighting object detection algorithms. It is hard to balance accuracy and efficiency, leading to trade-offs where improved accuracy in two-stage detectors compromises their applicability in real-time scenarios [7]. Moreover, many lightweight models sacrifice localization precision, particularly for small objects, limiting their effectiveness in practical applications.

Training deep object detection models requires large amounts of data. Currently, major UAV image object detection datasets include VisDrone (Vision Meets Drones) [8], UAVDT (The Unmanned Aerial Vehicle Benchmark: Object Detection and Tracking) [9], etc. The VisDrone dataset, for example, comprises images captured by multiple UAVs from oblique aerial perspectives, spanning landscapes across 14 Chinese cities. It includes 10,000 images and 2.6 million annotations, posing substantial challenges for object detection and tracking tasks. Images in the VisDrone dataset reach resolutions of up to  $2000 \times 1500$  pixels and encompass 10 distinct object categories. Notably, distinguishing between classes such as people and pedestrians proves challenging due to varying scales, orientations, uneven intensities, and significant image degradation. Addressing these challenges often necessitates incorporating attention mechanisms to better discern differential information among features.

When directly applying general object detection algorithms to UAV image target detection, the detection performance is typically significantly degraded due to the specific characteristics of UAV images. Consequently, researchers have undertaken targeted improvements, primarily focusing on optimizing two-stage detection algorithms, performing data augmentation [10], optimizing anchor-free methods [11,12], and optimizing lightweight models. In order to fully exploit the advantages of two-stage networks in detecting small objects, Cai et al. [13] addressed the issue of improving the IoU training threshold and proposed a network structure called Cascade R-CNN, which employs cascade-guided IoU resampling. This significantly enhances the accuracy of small-object detection but leads to a decrease in inference speed. Leveraging the characteristics of UAV image object aggregation, researchers proposed ClusDet [14], a multi-stage clustering detection network based on an improved R-CNN algorithm. This innovative framework integrates techniques like region clustering, slice detection, and scale adaptation, tailored to boost inference speed and enhance small-object detection rates in two-stage object detection networks applied to high-resolution UAV imagery. Moreover, employing training methodologies such as scaling for small objects and balancing positive–negative samples aims to elevate the accuracy of two-stage R-CNN models. However, these approaches face challenges in seamlessly handling scenes featuring a mix of small-scale and large-scale objects simultaneously.

Recently developed anchor-free networks are well suited for small-object detection in UAV imagery. For instance, CenterNet focuses on object center localization and offset prediction, which effectively enhances the detection rate of small objects [15]. However, the reliance on high-resolution feature maps can compromise the real-time performance of CenterNet. To address this, Google introduced MobileNet [16], which replaces traditional convolutions with depthwise separable convolutions, significantly reducing computational load and making it suitable for edge devices. Following this, various lightweight networks, such as EfficientDet [17] and GhostNet [18], have been proposed. Additionally, techniques

like L1 regularization-based model pruning and inter-group model distillation have gained traction for accelerating inference speed while maintaining efficiency on edge computing devices. However, these methods often incur a notable reduction in accuracy, posing a challenge for applications that require precise object detection [19].

Consider the large models: The Detection Transformer with Improved Denoising Anchor Boxes (DINO) enhances performance and efficiency over previous models by utilizing a denoising training approach through contrastive learning, a hybrid query selection method for anchor initialization, and a look-ahead-twice scheme for box prediction [20]. Additionally, RSPrompter [21] is a prompt learning method tailored for instance segmentation in remote-sensing imagery, leveraging the SAM foundational model. RSPrompter aims to generate prompt inputs for SAM, allowing it to automatically produce semantic instance-level masks. However, despite the advancements in large models for detection, existing methods still fail to effectively address the challenges posed by multi-scale scenes, leading to difficulties in accurately detecting and segmenting objects of varying sizes and contexts. At the same time, the large model weight also occupies a large amount of GPU memory, which is not conducive to the real-time flight of UAVs. If it is deployed locally, there is a problem of transmission bandwidth, which reduces the real-time detection.

It can be seen that the above algorithms have problems such as unbalanced network depth and width, insufficient classification accuracy, and insufficient inference speed, especially in complex detection scenes with large-scale spans. To solve these issues above, by fusing the characteristics of UAV images with the real-time performance and accuracy of one-stage YOLO series algorithms, this paper fully leverages the advantages of YOLOv8n to address issues such as imbalanced depth and width and insufficient classification accuracy, effectively improving the accuracy of real-time detection of small models in UAV scenarios. The main innovations of this study include the following:

- (1) Fusing the LSKA with SPPF of the backbone end (SPPF-LSKA) innovatively to perform weighted processing on multi-scale feature maps, which can better prioritize relevant features at different scales.
- (2) The neck being upsampled on the attention scale using ASF-DySample fusion of high-dimensional information from deep feature maps with detailed information from shallow ones.
- (3) Employing iRMB-DRB for C2f transformation leverages the synergies between dynamic global modeling and static local information fusion, leading to more comprehensive feature representations.
- (4) Replacing the original CIoU loss function with WIoUv3 can optimize the weighting for small objects to improve detection performance, focusing on anchor boxes of ordinary quality, enhancing the model's localization performance, which further proves the generalization of the proposed method.

The pipeline of the improved YOLOv8 algorithm is given in this paper, and the related work of UAV object detection and YOLOv8 is introduced. Then, different improvements including SPPF-LSKA, ASF-DySample, C2f-iRMB-DRB, and WIoUv3 are introduced in detail. Finally, the proposed method is evaluated by ablation experiments and comparison experiments.

## 2. Related Works

### 2.1. UAV Object Detection Workflow

The workflow for UAV-based object detection primarily encompasses four critical steps: data acquisition, preprocessing, model training and inference, and postprocessing. Initially, drones equipped with high-resolution cameras or other sensors operate within the target area, capturing a substantial volume of imagery data in real time. These data typically include images under various complex backgrounds and diverse lighting conditions. Subsequently, the acquired images undergo preprocessing, which involves image enhancement, noise reduction, and scale normalization to improve data quality and reduce computational load.

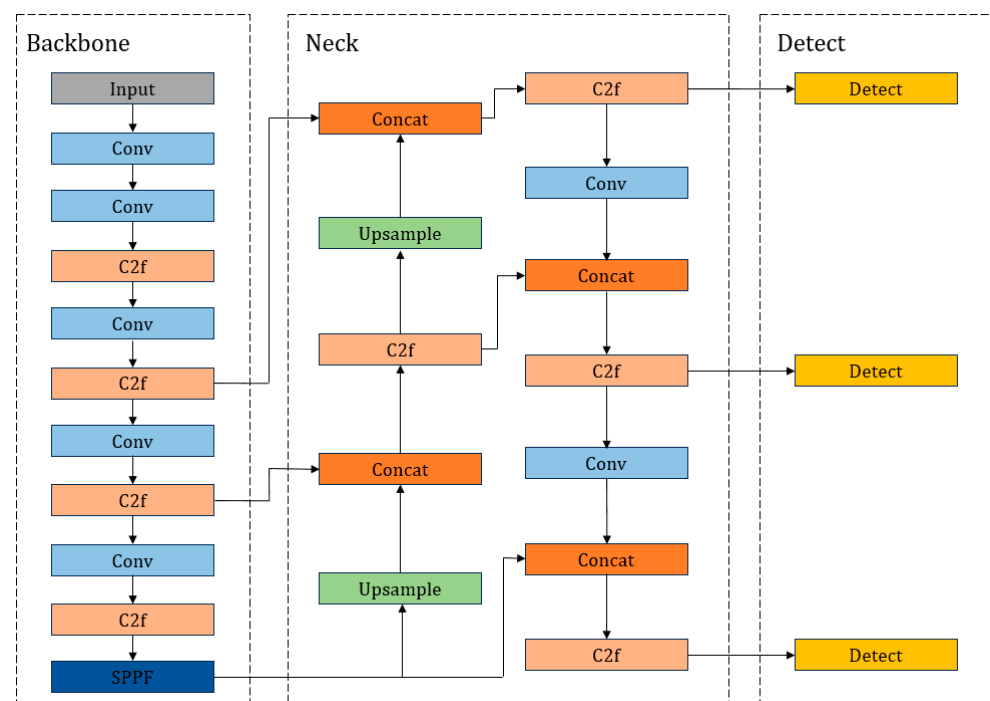
The next phase is model training and inference. For object detection tasks, deep-learning models such as Faster R-CNN and YOLO are widely employed. These models, trained on large-scale annotated datasets, can autonomously extract image features and predict the positions and classes of objects. During the training process, data augmentation techniques are applied to enhance the model's generalization capability, and optimization algorithms like Stochastic Gradient Descent (SGD) or Adam are used to adjust the model parameters [19].

Upon completion of model training, the model is deployed onto embedded systems or edge computing devices on the drone for real-time inference, enabling the rapid detection of objects in newly acquired images. The final step is postprocessing, where detection results are filtered and optimized. Techniques such as Non-Maximum Suppression (NMS) are employed to eliminate redundant bounding boxes, and object tracking algorithms are incorporated to enhance the continuity and stability of detections.

This entire workflow is iteratively refined to continuously improve the accuracy and real-time performance of UAV object detection, thereby meeting the demands of various application scenarios.

## 2.2. The YOLOv8 Algorithm

The following section will discuss the feasibility of applying YOLOv8 for UAV object detection. The YOLO algorithms are widely employed in object detection tasks involving small objects due to their advantages of high inference speed, real-time capability, and simplicity in network structure. YOLOv8 further optimizes the network architecture compared to its predecessors, enhancing the overall detection performance [22–24]. The network architecture of YOLOv8 consists of four parts: the input layer, the backbone module, the feature enhancement module, and the output layer [25]. This architecture is illustrated in Figure 1. After careful comparison, YOLOv8 is selected as the improved and optimized baseline in this paper, and details will be presented in the following sections.



**Figure 1.** YOLOv8 network structure.

### 2.2.1. Backbone

The backbone is primarily used for feature extraction. YOLOv8 replaces the Cross-Stage Partial (CSP) module in YOLOv5 with the lightweight C2f module, enhancing

feature representation capability through dense residual structures [26]. In YOLOv8, the C2f module is a residual block, with its core idea centered around the introduction of residual connections, enabling the network to learn a direct mapping relationship between input and output, thereby optimizing the training process. The C2f module consists of two convolutional layers (Conv), interconnected via residual connections. In this residual connection, the input signal is directly passed to the output signal while concurrently learning the mapping relationship between the input and output through convolutional operations. This structure effectively addresses the issues of vanishing gradients and limited representational capacity commonly encountered in deep neural networks. The primary function of the C2f module in YOLOv8 is to enhance feature extraction capabilities. By incorporating residual connections, the C2f module allows the network to better learn and leverage the relational information between features, thus improving the accuracy of feature extraction. Furthermore, due to the relatively simple structure of the C2f module, it can effectively reduce the computational burden and complexity of the model when computational resources are limited, thereby enhancing the operational efficiency of the model.

In addition, the SPP module (spatial pyramid pooling) [27], originally used in earlier YOLO versions like YOLOv5, has been replaced with the Spatial Pyramid Pooling Fast (SPPF) module [28]. The SPPF module replaces the original three different-sized convolutional kernels with three  $5 \times 5$  convolutional kernels. This substitution is because concatenating two  $5 \times 5$  convolutional kernels is equivalent to using one  $9 \times 9$  convolutional kernel, and similarly, concatenating three  $5 \times 5$  convolutional kernels is equivalent to using one  $13 \times 13$  convolutional kernel. Compared to using larger convolutional kernels directly, concatenating multiple smaller kernels reduces network's computational load, thereby improving inference speed for UAV objects.

### 2.2.2. Neck

The Neck is primarily utilized for feature fusion, employing the Path Aggregation Network (PAN) [29] in conjunction with the C2f module. It facilitates the fusion of feature maps at different scales from the three stages of the backbone, aiding in the aggregation of shallow information into deeper features.

YOLOv8's neck layer continues the FPN+PAN [30] concept from YOLOv5. Compared to YOLOv5, YOLOv8 replaces C3 with the C2f module and removes the convolutional layer before upsampling, directly upsampling output features from different stages of the backbone feature extraction network. These modifications further optimize the network structure, enhancing detection efficiency.

### 2.2.3. Head

From YOLOv3 to YOLOv5, the detection head has consistently been coupled, meaning that a single convolutional layer is used to simultaneously perform both classification and localization tasks. It was not until the introduction of YOLOX [31] that the YOLO series first adopted a decoupled head structure.

YOLOv8 also employs a decoupled head, with two parallel branches extracting category features and position features separately. Subsequently, each branch utilizes a  $1 \times 1$  convolutional layer to perform UAV object classification and localization tasks.

### 2.2.4. Loss

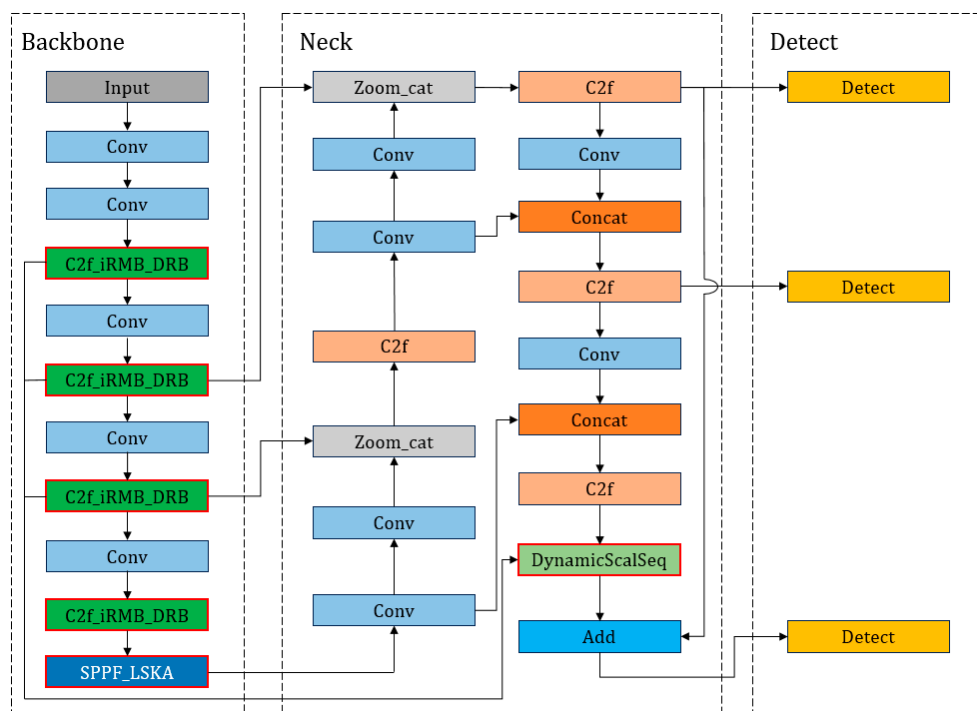
The loss function in YOLOv8 consists primarily of two components: classification loss and localization loss. For YOLOv8, its classification loss is represented by VFL loss (varifocal loss), while its regression loss takes the form of CIoU loss and DFL loss [32]. The varifocal loss is defined as follows [33]:

$$VFL(p, q) = \begin{cases} -q(q \log(p) + (1 - q)(1 - p)) & q > 0 \\ -\alpha p^\gamma \log(1 - p) & q = 0 \end{cases} \quad (1)$$

where  $p$  represents the predicted class score,  $p \in [0, 1]$ ;  $q$  represents the predicted target score (if it is the true class,  $q$  is the predicted and true IoU; if it is another class,  $q$  is 0); and  $\alpha$  and  $\gamma$  represent the hyper-parameters of varifocal loss. VFL loss utilizes asymmetric parameters to weight positive and negative samples differently, reducing only the negative samples to achieve equal treatment of foreground and background contributions to the loss.

### 3. Methods

This section describes the improvements employed in the YOLOv8 baseline, with Figure 2 illustrating the structural attributes of the augmented YOLOv8 model. Firstly, the method encompasses three components: SPPF-LSKA, ASF-DySample, and C2f-iRMB-DRB, as is shown in Figure 2. Then, the IoU (Intersection over Union) mechanism transitions from the default CIoU to WIoUv3.



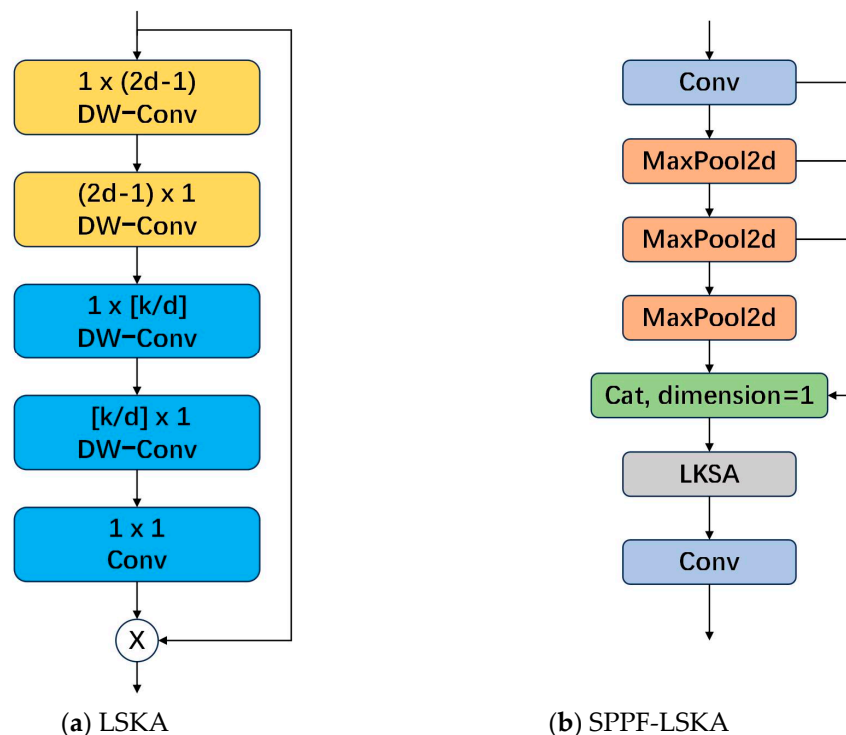
**Figure 2.** Model architecture of YOLO-DroneMS. SPPF-LSKA is adopted to perform weighted processing on multi-scale feature maps, while C2f-iRMB-DRB is used for balancing the advantages of dynamic global modeling and local information fusion. Additionally, ASF-DySample is utilized to dynamically upsample the neck section.

#### 3.1. SPPF-LSKA

The existing SPPF (Spatial Pyramid Pooling Factor) is a crucial component designed for extracting multi-scale features. Utilizing spatial pyramid pooling, SPPF merges feature maps of various scales, thereby enhancing the model’s robustness to changes in target scales. However, despite its ability to enhance the extraction of multi-scale features to some extent, there are still some defects in SPPF: Although SPPF utilizes spatial pyramid pooling to handle multi-scale features, this simplistic fusion approach may not fully exploit the correlations among features at different scales. Due to the design constraints of spatial pyramid pooling, SPPF may not effectively handle targets at very small or very large scales. Additionally, SPPF may encounter performance bottlenecks when dealing with dense scenes.

To further enhance the multi-scale feature extraction capability, researchers have introduced Large Separable Kernel Attention (LSKA) [34]. LSKA is a form of attention mechanism based on depthwise separable kernels, aimed at capturing dependencies be-

tween spatial and channel domains to increase the model's sensitivity to target features (Figure 3a). The Visual Attention Network (VAN) incorporating Large Kernel Attention (LKA) modules has demonstrated superior performance across various visual tasks [35], surpassing traditional visual transformers.



**Figure 3.** Structure of LSKA and SPPF-LSKA. The LSKA mechanism is applied after SPPF to perform weighted processing on multi-scale feature maps.

However, the deep convolutional layers within these LKA modules incur quadratic increases in computation and memory consumption as the kernel size expands. To address these challenges and enable the use of significantly larger kernels within attention modules of the VAN, a series of Large Separable Kernel Attention modules, collectively termed LSKA, are proposed.

As the kernel size expands, the LSKA approach subtly steers VAN towards prioritizing object shapes over textures. This inclination towards shape features offers distinct benefits in object detection scenarios, where resilience to texture and lighting variations holds paramount importance. In YOLO-DroneMS, the LSKA mechanism is strategically employed post-SPPF to dynamically process multi-scale feature maps with weighted attention (Figure 3b).

Specifically, LSKA first calculates spatial and channel weights for each feature map, and then adjusts the feature maps accordingly based on these weights. This allows the model to focus more on features relevant to the target, thereby enhancing detection accuracy. By leveraging LSKA within SPPF, YOLO-DroneMS can better prioritize relevant features at different scales, thereby improving its overall detection accuracy without significantly increasing computational overhead.

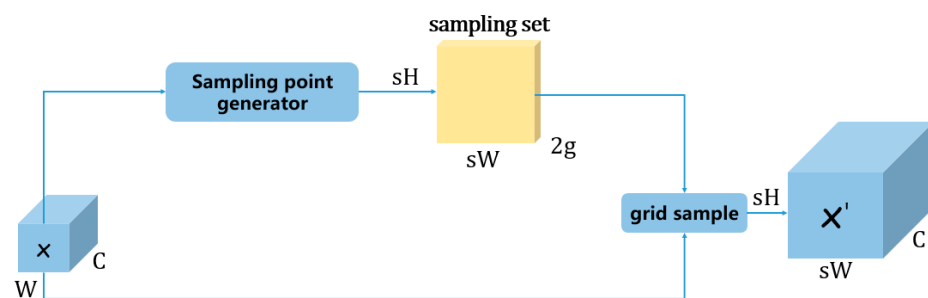
### 3.2. ASF-DySample

YOLOv8, like many object detection models, struggles with effectively fusing features from different scales. Traditional upsampling methods, such as dynamic convolution-based methods like CARAFE and FADE [36], can be computationally expensive. To solve these two problems, this study proposes a novel network structure of ASF-DySample to

transform the neck of YOLO-DroneMS, and this section will introduce the improvement in detail.

Attentional Scale Sequence Fusion (ASF) [37] is a novel approach for scale-aware feature fusion, which better integrates the high-dimensional information from deep feature maps with the detailed information from shallow ones. Here, the size of the image changes, but scale-invariant features occur during image downsampling. Scale space is constructed along the axis of scale in the image, representing not only scale but also the range of scales that objects can possess. Scale refers to the level of detail in an image. While a blurred image may lose details, the structural features of the image can be preserved.

On the other hand, DySample is a novel dynamic upsampling method that achieves upsampling through learned sampling [38]. This approach, when employed for upsampling in image or video processing, circumvents the high complexity and computational burden associated with traditional dynamic convolution-based upsampling methods. Implemented from the perspective of point sampling, DySample dynamically adjusts sampling points by summing offsets with original grid positions (Figure 4). This method not only exhibits high resource efficiency but can also be readily implemented using standard PyTorch built-in functions. Such a design allows DySample to perform well across various dense prediction tasks, including semantic segmentation, object detection, instance segmentation, panoptic segmentation, and monocular depth estimation.



**Figure 4.** Point sampling of DySample. DySample dynamically adjusts sampling points by summing offsets with original grid positions.

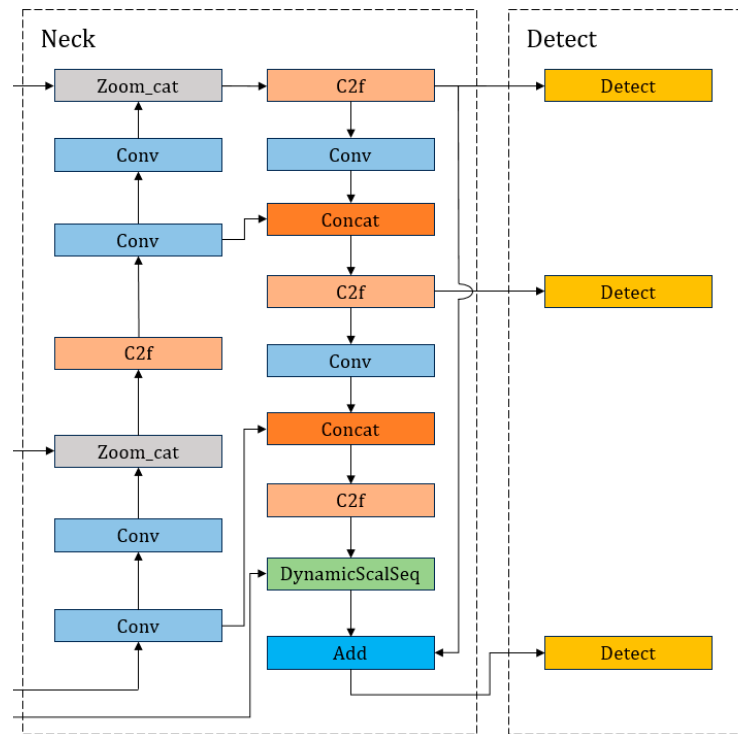
DySample's dynamic sampling reduces the accumulation of prediction errors by controlling offset ranges, a crucial aspect in enhancing prediction accuracy near boundaries [39]. Through methods such as improving initial sampling positions and adjusting offset ranges, DySample effectively enhances the performance and quality of upsampling.

ASF-DySample transforms the neck part of YOLOv8 by introducing the DynamicScalSeq module and integrating it into the feature fusion process (Figure 5). The DynamicScalSeq module takes as input the features from different scales and dynamically samples them according to learned weights. This module dynamically adjusts the sampling strategy based on the features' scale, allowing for better adaptation to the feature of the input image.

At the end of the neck, the output of the DynamicScalSeq module is merged with the features from the shallow feature maps using the Add operation. This integration enhances the fusion of high-dimensional information from deep feature maps with detailed information from shallow feature maps, contributing to improved object detection performance.

Overall, ASF-DySample introduces a dynamic upsampling mechanism through the DynamicScalSeq module, which optimizes the fusion of features from different scales in the YOLOv8 neck architecture, leading to more effective object detection.

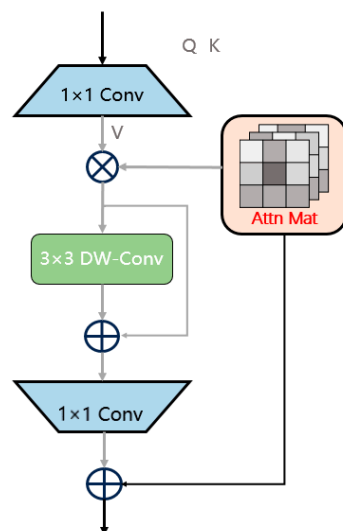




**Figure 5.** Architecture of the ASF-DySample, in which the DynamicScalSeq module merges the features from shallow feature maps by the Add operation.

### 3.3. C2f-iRMB-DRB

The main idea of the Inverted Residual Mobile Block (iRMB) [40] is to fuse lightweight CNN architectures and attention-based model structures (similar to ACmix) to create efficient mobile networks. By rethinking the components of inverted residual blocks (IRBs) [41] and transformers [42], iRMB achieves a unified perspective, thereby extending CNN’s IRB to attention-based models (Figure 6).



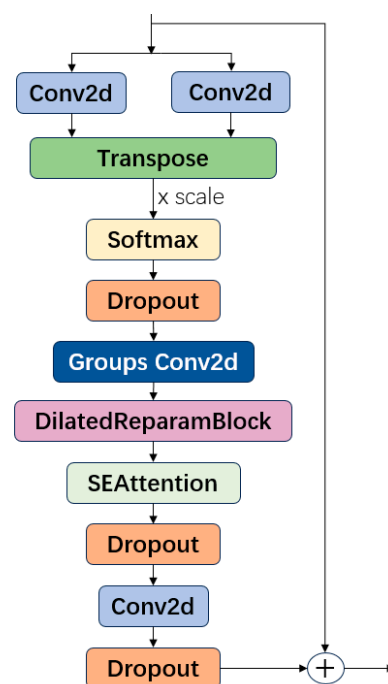
**Figure 6.** Structure of iRMB network. iRMB fuses lightweight CNN architectures and attention-based model structures.

The design goal of iRMB is to effectively utilize computational resources and achieve high accuracy while keeping the model lightweight. The primary innovation of the iRMB module lies in the lightweight characteristics of Convolutional Neural Networks (CNNs)

and the dynamic processing capabilities of transformer models. This structure is particularly suitable for dense prediction tasks on mobile devices, as it aims to provide efficient performance in environments with limited computational capabilities.

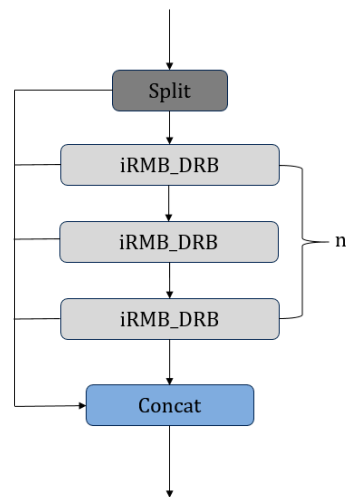
iRMB improves the handling of information flow through its inverted residual design, allowing for the capture and utilization of long-range dependencies while maintaining model lightweightness, which is crucial for tasks such as image classification, object detection, and semantic segmentation. This design enables the model to operate efficiently on resource-constrained devices while maintaining or enhancing prediction accuracy.

The DilatedReparamBlock in UniRepLKNet [43] enhances the model's ability to handle larger receptive fields by employing dilated convolutions, thereby improving performance on complex or fine-grained tasks (Figure 7). This enables the model to better capture details and contextual information in images, which is particularly beneficial for image-processing tasks.



**Figure 7.** Structure of the iRMB-DRB network.

By incorporating DRB into the iRMB and integrating it into the C2f architecture, the model gains enhanced ability to capture fine details and contextual information from images, as is shown in Figure 8. This is crucial for tasks requiring precise object detection and recognition. The integration of DRB within iRMB enables efficient attention mechanisms and spatial information processing, facilitating effective feature fusion within the baseline architecture. This ensures better integration of features across different scales, leading to more comprehensive feature representations. The iRMB-DRB approach enhances the model's capability to handle features at various scales. This is essential for object detection tasks, where objects may appear at different sizes within an image. By effectively processing features at different scales, the model improves its overall performance and accuracy in detecting objects of varying sizes and complexities.



**Figure 8.** Structure of the C2f-iRMB-DRB submodule. The iRMB-DRB network enhances the model’s capability to handle features at various scales.

### 3.4. *WIoU Loss*

The loss function of YOLOv8 comprises confidence loss, classification loss, and bounding box loss functions, with the bounding box loss function reflecting the error between the ground truth boxes and predicted boxes. The design of the bounding box loss function significantly impacts the performance of object detection, as a good bounding box loss function can enhance the precision of UAV object detection. Object detection datasets frequently contain instances of suboptimal quality, and an excessive focus on bounding box regression for such cases may impede advancements in UAV object detection capabilities.

To address this issue, the Wise-IoU (*WIoU*) loss function is proposed [44], which is a loss function with bounding box localization based on a dynamic non-monotonic focusing mechanism. For low-quality samples, geometric metrics such as aspect ratio and distance between predicted and ground truth boxes increase the penalty, affecting the model’s generalization ability. When the predicted boxes align well with the ground truth boxes, the penalty on geometric metrics should be reduced. *WIoU* has three versions: v1 constructs an attention-based bounding box loss, while v2 and v3 add a focusing mechanism on top of v1, with v3 demonstrating superior performance.  $WIoU_{v1}$  is constructed based on distance metrics, as presented in Equations (2) and (3) [45].

$$\mathcal{R}_{WIoU} = \exp\left(\frac{(x - x_{gt})^2 + (y - y_{gt})^2}{(W_g^2 + H_g^2)^*}\right), \quad (2)$$

$$\mathcal{L}_{WIoU_{v1}} = \mathcal{R}_{WIoU} \mathcal{L}_{IoU}, \quad (3)$$

where  $x$  and  $y$  represent the center coordinates of the bounding box,  $x_{gt}$  and  $y_{gt}$  represent the center coordinates of the target box,  $W_g$  and  $H_g$  represent the width and height of the minimum bounding box,  $WIoU \in [1, e]$  expands the *IoU* (Intersection over Union) loss function value *IoU* for regular quality anchor boxes to a certain extent, while  $IoU \in [0, 1]$  significantly reduces the *WIoU* for high-quality anchor boxes. When there is a high overlap between the target box and the anchor box, emphasis is placed on the distance between the centers of the two boxes. The asterisk (\*) denotes the detachment of  $W_g$  and  $H_g$  from the computational graph.

The  $WIoU_{v2}$  bounding box regression loss function is designed to reduce the contribution of simple samples to the loss value, while enabling the model to focus on the

monotonic focusing coefficient of difficult samples, thereby enhancing the UAV object detection performance. The formula for the  $WIoU_{v2}$  loss function is shown in Equation (4) [43].

$$\mathcal{L}_{WIoU_{v2}} = \mathcal{L}_{IoU}^{\gamma^*} \mathcal{L}_{WIoU_{v1}}, \gamma > 0 \quad (4)$$

where  $\gamma^* IoU$  decreases as  $IoU$  decreases during training, which can lead to slow convergence in the later stages of model training. To address this issue, a moving-average  $IoU$  is introduced to maintain the overall  $\gamma^* IoU / IoU$  at a relatively high level, as shown in Equation (5).

$$\mathcal{L}_{WIoU_{v2}} = \left( \frac{\mathcal{L}_{IoU}^*}{\mathcal{L}_{IoU}} \right)^\gamma \mathcal{L}_{WIoU_{v1}} \quad (5)$$

where  $\gamma$  is the exponential factor. The  $WIoU_{v3}$  loss function utilizes outlierliness to describe the quality of anchor boxes, where lower outlierliness indicates higher-quality anchor boxes, while higher outlierliness suggests lower-quality anchor boxes. The definition of outlierliness is shown in Equation (6).

$$\beta = \frac{\mathcal{L}_{IoU}^*}{\mathcal{L}_{IoU}} \quad (6)$$

For anchor boxes with low outlierliness, a small gradient boost is assigned to garner more attention to ordinary anchor boxes. Conversely, for anchor boxes with high outlierliness, a small gradient boost is assigned to prevent significant harmful gradients from low-quality anchor boxes. This constructs a focusing coefficient applied to  $WIoU_{v1}$  to derive  $WIoU_{v3}$ , as depicted in Equation (7) [46].

$$\mathcal{L}_{WIoU_{v3}} = r \mathcal{L}_{WIoU_{v1}}, r = \frac{\beta}{\delta \alpha^{\beta - \delta}} \quad (7)$$

where  $\alpha$  and  $\delta$  are hyper-parameters,  $\beta$  is the non-monotonic focusing coefficient, and  $r$  is the conversion coefficient. Due to the dynamic nature of  $IoU$  and the quality partitioning criteria for anchor boxes,  $WIoU_{v3}$  can dynamically allocate gradients tailored to the circumstances of each moment, thereby enhancing model performance for UAV object detection.

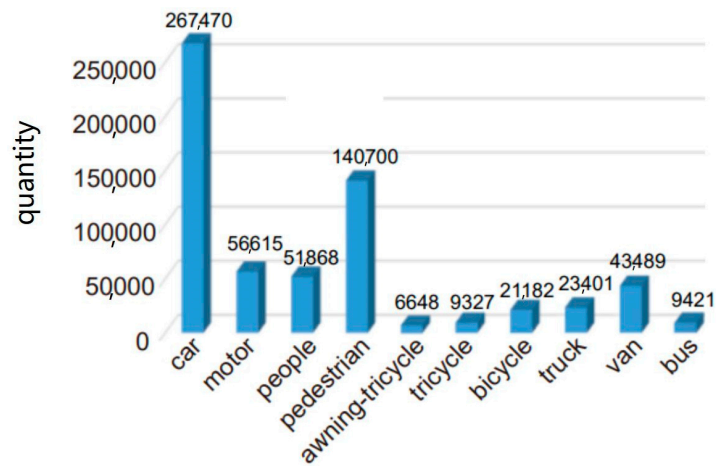
## 4. Experimental Studies

### 4.1. Dataset

This study employs the VisDrone2019 dataset [8] for experimental validation. Both the training and validation datasets are partitioned into images sized  $800 \times 800$  with a stride of 600. The training dataset comprises 25,447 annotated images, while the validation dataset includes 1115 images with corresponding annotations. From the validation set, 547 images are reserved for testing purposes. The VisDrone2019 dataset encompasses 10 classes, such as pedestrian, people, bicycle, car, van, truck, tricycle, awning-tricycle, bus, and motor. Through numerical analysis, as detailed in Table 1, the dataset reveals a class imbalance, characterized by numerous small objects and fewer large objects. Additionally, certain classes exhibit minimal variability and significant inter-class confusion, presenting formidable challenges (Figure 9).

**Table 1.** Statistics of different size types of objects of the VisDrone2019 dataset.

Object Type	Small ( $0 \times 0 \sim 32 \times 32$ )	Mid ( $32 \times 32 \sim 96 \times 96$ )	Large ( $96 \times 96 \sim$ )
Quantity ( $\times 10,000$ )	44.44	18.63	1.704



**Figure 9.** Total number of object instances in the VisDrone2019 dataset.

#### 4.2. Experimental Configuration

The construction, training, and testing of each UAV object detection model in the experiment are conducted within the PyTorch deep-learning framework, utilizing CUDA and CuDNN for acceleration. As shown in Table 2, the experimental setup is based on the Linux operating system, with an Intel(R) Xeon(R) Gold 6130 CPU, 25 GB of RAM, and an NVIDIA GeForce RTX 3090 Ti GPU with 24 GB of VRAM. PyTorch version 2.0.0 and Python version 3.8.5 are employed.

**Table 2.** Experimental configuration.

Name	Configure
Operating System	Linux
CPU	Intel(R) Xeon(R) Gold 6130
GPU	NVIDIA GeForce RTX 3090 Ti
GPU Memory	24 G
Programming language	Python 3.8.5
Deep-learning framework	Pytorch 2.0.0
Name	Configure

#### 4.3. Evaluation Indicators

To validate the model performance, various object detection evaluation metrics are employed, including precision, recall, average precision (*AP*), and mean average precision (*mAP*). Recall represents the model's ability to retrieve all relevant instances, assessing the model's completeness in object detection, while precision evaluates the model's accuracy in predicting positives. The formulas for precision and recall are shown in Equations (8) and (9), respectively [47].

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

$$Recall = \frac{TP}{TP + FN} \quad (9)$$

where *TP* denotes the number of true positives predicted by the model, *FP* represents the number of false positives (actual negatives incorrectly predicted as positives), and *FN* represents the number of false negatives (actual positives incorrectly predicted as negatives).

Using recall and precision for the same class as the horizontal and vertical axes, respectively, the resulting curve is termed the precision–recall (P-R) curve. The area under the P-R curve, bounded by the axes, represents the average precision (*AP*) for that class.

The  $mAP$  is obtained by averaging  $AP$  values across all classes [48]. The formulas for calculating  $AP$  and  $mAP$  are presented in Equations (10) and (11), respectively.

$$AP = \int_0^1 p(r)dr, \quad (10)$$

$$mAP = \sum_{i=1}^N AP_i / N \quad (11)$$

where  $N$  represents the total number of classes in the training set, and  $i$  denotes the  $i$ -th class. In this study,  $N$  is set to 10, as the number of classes in the VisDrone2019 dataset is 10.

## 5. Results

### 5.1. Ablation Experiments

To validate the performance gains brought by three optimization strategies—WIoUv3, SPPF-LSKA, ASF-DySample, and iRMB-DRB—this study conducted the following ablation experiments on the VisDrone2019 dataset. The experimental results are shown in Table 3.

**Table 3.** Ablation experiment results of modules on the VisDrone2019 dataset.

Models	WIoUv3	SPPF-LSKA	ASF-DySample	iRMB-DRB	F1-Score	Precision	Recall	mAP@50	Params/M
YOLOv8n					0.32	1	0.51	0.276	3.15
	✓				0.33	1	0.51	0.279	3.15
	✓	✓			0.32	1	0.51	0.280	3.43
	✓	✓	✓		0.33	1	0.51	0.281	3.31
	✓	✓	✓	✓	0.35	1	0.52	0.312	3.14

Across all improvements of the YOLOv8n model, precision remains consistently high at 1, indicating minimal false-positive predictions. The F1-score shows a slight variation ranging from 0.32 to 0.35, suggesting a stable overall performance.

Recall also exhibits consistency, hovering around 0.51 to 0.52, indicating the model's ability to capture a significant portion of positive instances. Notably, the mean average precision (mAP@50) demonstrates a gradual improvement from 0.276 in the baseline YOLOv8n model to 0.312 in the fully modified iteration. YOLOv8n-WIoUv3 introduces the WIoUv3 component, resulting in a marginal increase in F1-score, recall, and mAP@50 compared to the baseline.

Further enhancements with SPPF and LSKA contribute to a minor improvement in mAP@50. Additional modifications with ASF and dynamic sampling lead to a slight boost in mAP@50. The incorporation of iRMB and DRB yields the most significant improvement in mAP@50, reaching 0.312. The progressive inclusion of enhancements results in incremental improvements in object detection performance. Finally, the last iteration incorporating iRMB and DRB achieves the highest mAP@50 score, indicating the effectiveness of these modifications in refining the model's accuracy. And it is noted that the number of parameters of YOLO-DroneMS obtained after the improvement is slightly lower than that of YOLOv8.

### 5.2. Comparison of Models

To validate the effectiveness of the proposed method, this study utilized the YOLOv8n model as the baseline algorithm and conducted comparative experiments, including SSD, Faster R-CNN, Cascade R-CNN, YOLOv3, and YOLOv5 for validation, as shown in Table 4. It presents a comparative analysis of various models' performance on the VisDrone2019 dataset, focusing on different object categories, including pedestrian, people, bicycle, car, van, truck, tricycle, awning-tricycle, bus, and motor, along with the metrics mAP@50 and FPS.

**Table 4.** Results of different models on the VisDrone2019 dataset (%). mAP@50 and FPS are metrics (The bold denotes the best effect and highlights the algorithm we proposed).

Models	Pedestrian	People	Bicycle	Car	Van	Truck	Tricycle	Awning-Tricycle	Bus	Motor	mAP@50	FPS
SSD [49]	0.027	0.023	0.002	0.298	0.062	0.056	0.018	0.012	0.215	0.031	0.074	9.6
Faster R-CNN [50]	0.187	0.090	0.063	0.666	0.335	0.238	0.126	0.100	0.478	0.171	0.246	14.5
Cascade R-CNN [13]	0.166	0.072	0.066	0.677	0.362	0.255	0.133	0.103	0.491	0.169	0.249	9.3
YOLOv3 [51]	0.069	0.041	0.023	0.442	0.136	0.171	0.046	0.051	0.374	0.071	0.142	40.1
YOLOv5 [52]	0.222	0.138	0.057	0.650	0.213	0.178	0.071	0.084	0.457	0.203	0.227	68.9
YOLOv7 [53]	0.179	0.0974	0.0448	0.618	0.272	0.26	0.0935	0.111	0.433	0.185	0.229	66.1
YOLOv8n [23]	0.226	0.117	0.065	0.670	0.305	<b>0.337</b>	0.143	<b>0.149</b>	<b>0.518</b>	0.225	0.276	78.7
RT-DETR [54]	0.0773	0.0347	0.0171	0.386	0.179	0.172	0.0489	0.0633	0.306	0.078	0.248	73.7
YOLOv9 [55]	0.207	0.108	0.0517	0.643	0.287	0.291	0.107	0.123	0.472	0.211	0.25	80.8
Ground DINO 1.5 [56]	0.215	0.092	0.061	0.627	0.279	0.296	0.102	0.137	0.456	0.208	0.262	73.1
YOLOv10 [57]	0.19	0.125	0.059	0.63	0.271	0.262	0.0883	0.119	0.453	0.199	0.243	78.3
<b>YOLO-DroneMS</b>	<b>0.330</b>	<b>0.262</b>	<b>0.069</b>	<b>0.744</b>	<b>0.369</b>	0.265	<b>0.203</b>	0.106	0.423	<b>0.351</b>	<b>0.312</b>	<b>83.3</b>

Firstly, existing models like SSD and YOLOv3 demonstrate relatively lower precision across most classes compared to more advanced architectures such as Faster R-CNN, Cascade R-CNN, YOLOv5, YOLOv7 [53], YOLOv8n (Figure 1), and the latest models (RT-DETR [54], YOLOv9 [55], Ground DINO 1.5 [56], and YOLOv10 [57]). While Faster R-CNN and Cascade R-CNN exhibit competitive performance, particularly in detecting cars and related vehicles, they still show limitations in accurately identifying smaller objects like tricycles and awning-tricycles.

Secondly, YOLOv5 and YOLOv8n represent advancements in object detection, offering improved precision across multiple classes compared to their predecessors. However, they still struggle with certain object classes, such as tricycles and buses. SSD achieves an mAP@50 score of 0.074, indicating relatively lower overall performance, but Faster R-CNN demonstrates an mAP@50 score of 0.246, showing improved performance compared to SSD. The Cascade R-CNN slightly outperforms Faster R-CNN with an mAP@50 score of 0.249. Considering the YOLO series models, YOLOv3 shows a moderate performance with an mAP@50 score of 0.142, YOLOv5 exhibits competitive performance with an mAP@50 score of 0.227, and YOLOv8n shows further improvement with an mAP@50 score of 0.276, noting that the latest models RT-DETR and YOLOv9 do not substantially improve in mAP@50. Then, the improved model achieves the highest mAP@50 score of 0.312, indicating superior overall performance compared to the other models.

Thirdly, the final model achieves the highest mAP@50 score of 0.312, indicating its superior overall performance compared to the other models (Figure 10). Remarkably, the proposed model outperforms all other models across the board. It achieves significantly higher precision in detecting pedestrians, people, cars, and other objects, as evidenced by its superior performance in almost all object classes. The superior performance of the improved model suggests better robustness to variations in object size, orientation, and occlusion within the VisDrone2019 dataset. This robustness is particularly crucial in real-world scenarios where objects may vary significantly in appearance and context.

The inference speed of YOLOv8-DroneMS far surpasses that of the other models, including the latest SOTAs-Ground DINO 1.5 and YOLOv10, on mAP@50 and speed. For the top-performing YOLOv8n and YOLOv9, our model exhibits a significant improvement, being 5.8% faster than YOLOv8n and 2.7% faster than YOLOv9.

Finally, the confusion matrices generated by YOLOv8n and the YOLOv8-improved model are illustrated in Figure 11 below.

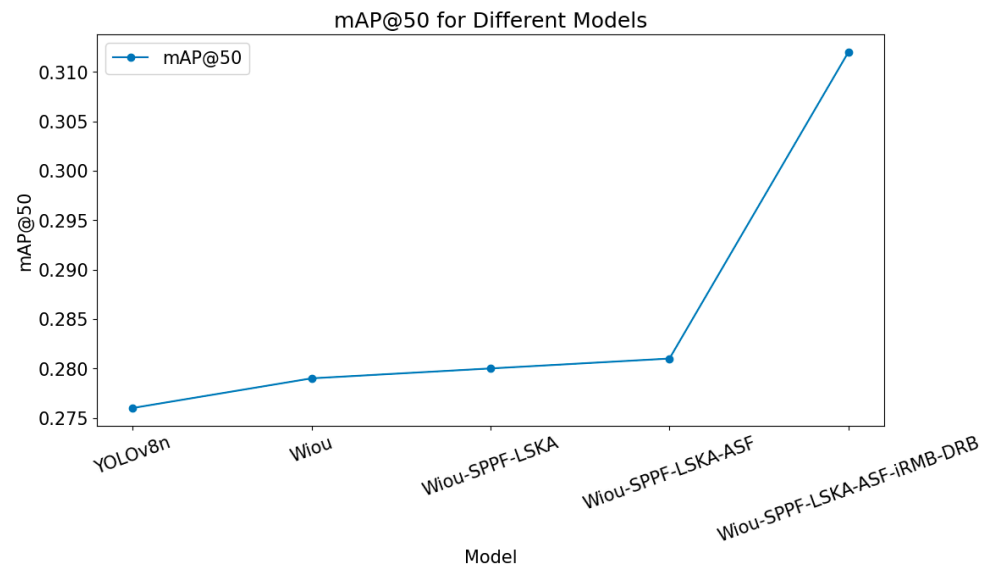
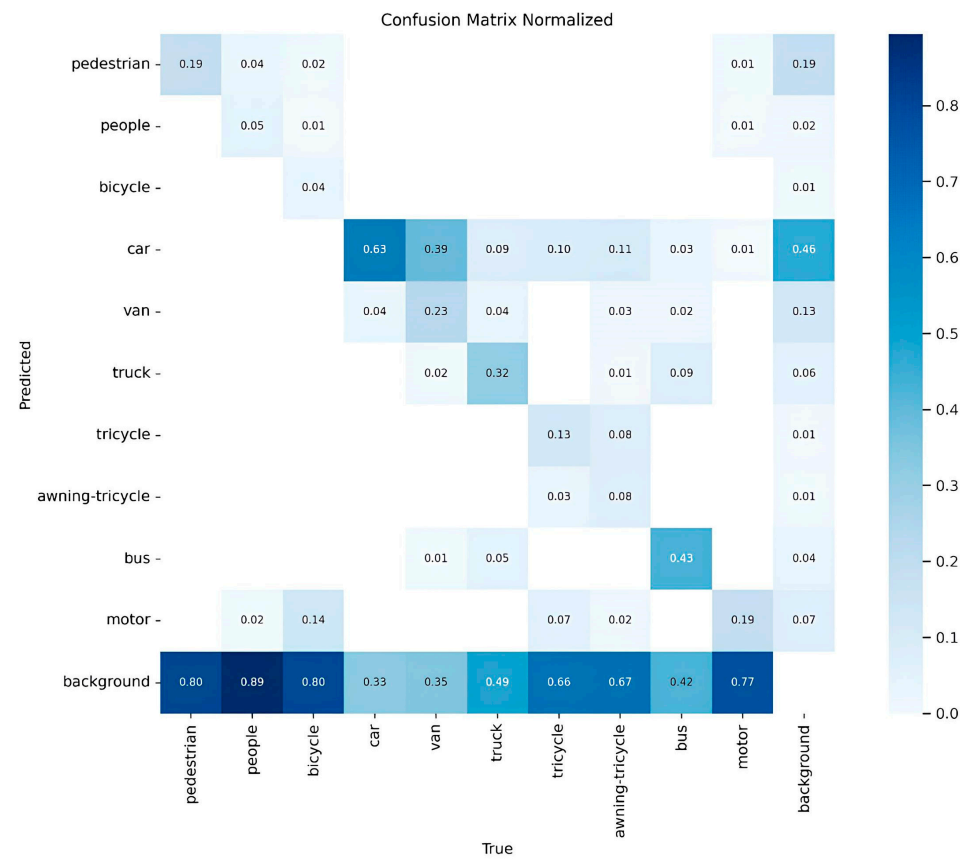


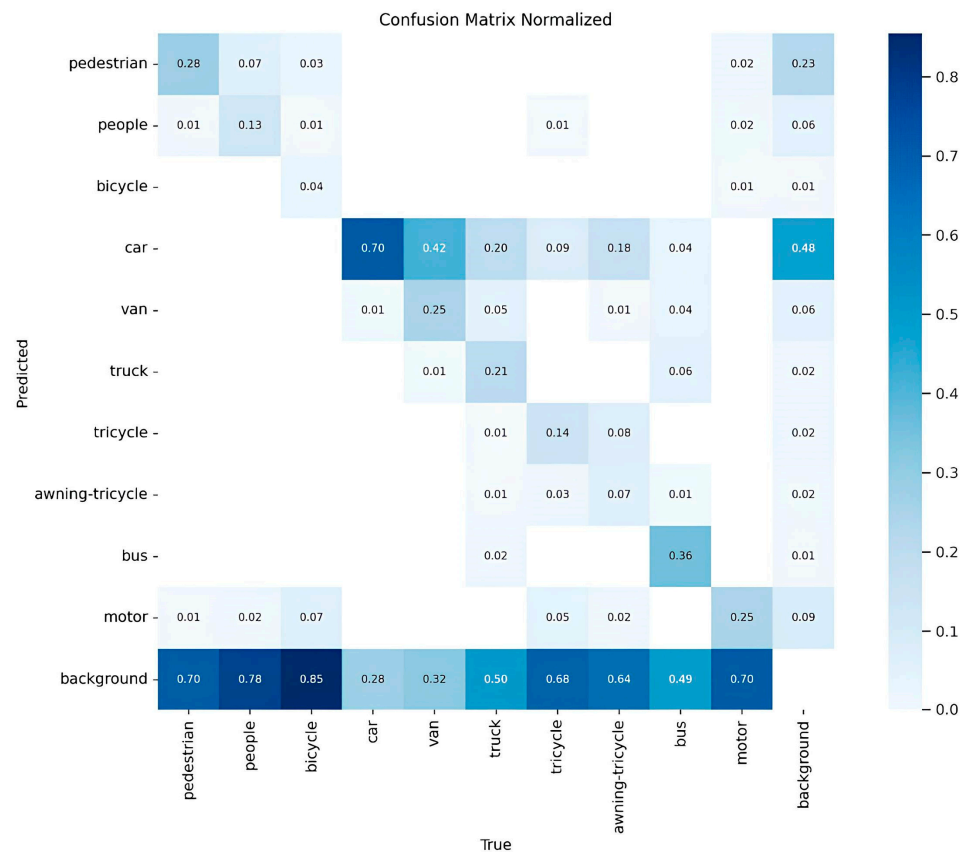
Figure 10. Comparison of mAP@50 for different models on the VisDrone2019 dataset.



(a) Confusion matrix of YOLOv8n.

Figure 11. Cont.





(b) Confusion matrix of YOLO-DroneMS.

**Figure 11.** Comparison of confusion matrices of YOLOv8 and YOLO-DroneMS. It is seen that the “pedestrian” class has an increase of 9%, and the “people” and “car” categories have improved by 8% and 7%, respectively.

The optimal model demonstrates a significant improvement in classification accuracy while reducing classification confusion. Particularly, the recognition accuracy for the “pedestrian” class shows the most notable enhancement, with an increase of 9%. Additionally, the “people” and “car” classes have improved by 8% and 7%, respectively. There is a general advancement in classification accuracy across other classes as well, with the “car” class achieving the highest classification accuracy of 70%, while the “bicycle” class exhibits the lowest at only 4%. This disparity reveals the model’s recognition preference for different categories.

Regarding the identification of small objects such as “motor”, the baseline YOLOv8n model achieves a classification accuracy of only 19%, with a 14% probability of misidentifying “motor” as “bicycle”. However, through optimization in our optimal model, the recognition accuracy for “motor” increases to 25%, representing a 6% improvement over YOLOv8n. YOLO-DroneMS notably enhances the accuracy of drones in identifying pedestrians, vehicles, and other small objects.

### 5.3. Visualization

In order to test the actual effectiveness of drone object detection, the following visualized heatmap analysis is conducted. As shown in Figure 12, SSD and Cascade are not focused enough on target recognition and exhibit significant missed detection cases. Faster R-CNN shows some improvement, but the performance is not as good as YOLOv5s, which is not sensitive enough to pedestrian targets. Ultimately, YOLOv8 and the proposed YOLO-DroneMS in this paper show better performance, with YOLO-DroneMS showing a slight improvement in target confidence compared to YOLOv8 and also demonstrating

better accuracy in detecting distant targets than YOLOv8. Compared to the latest SOTA like YOLOv10, YOLO-DroneMS has a clearer recognition of small and dense targets at the far end.

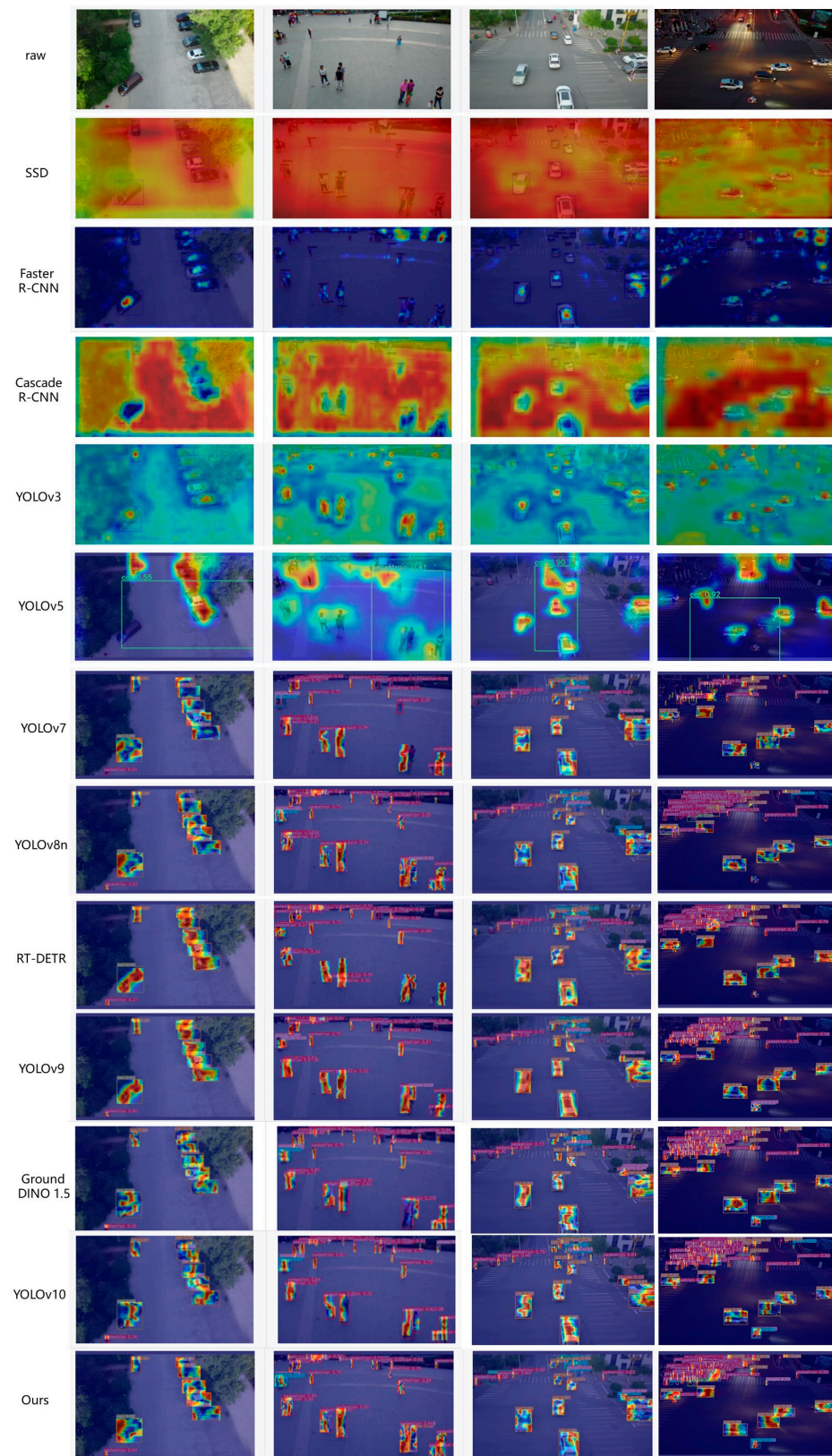
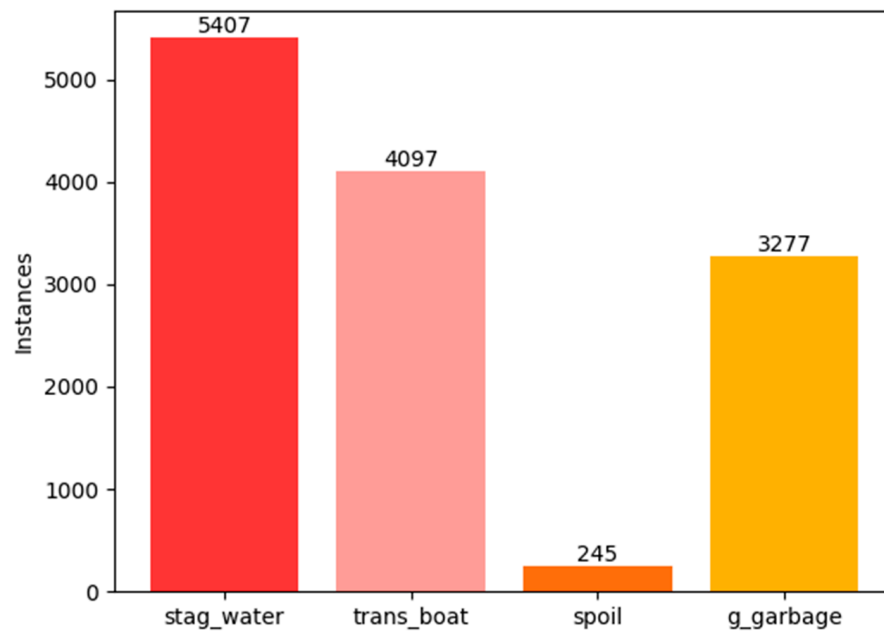


Figure 12. Visualization of the comparison of different models.

In summary, the YOLOv8-improved model not only enhances overall classification accuracy but also specifically optimizes the recognition capability for small objects, marking a significant advancement for complex drone vision tasks.

#### 5.4. Generalization Experiments

In order to thoroughly validate the effectiveness and robustness of the proposed method, this study conducted a comparative verification on the private remote-sensing dataset named RiverInspect-2024. And the next practical application of YOLO-DroneMS model is considered for a river drone-inspection project, where UAV object models are deployed for engineering monitoring during the construction phase of hydraulic engineering projects, as well as river monitoring during the operational phase of hydraulic engineering. This dataset comprises a total of 7182 images, randomly and evenly split into 5013 for the training set, 1441 for validation set, and 728 for testing set. And there are four object classes including stag-water, transportation boat, spoil, and garbage, as shown in Figure 13. When selecting the same reference size, stag\_water represents large-sized target objects, g\_garbage is biased towards medium-sized target objects, while trans\_boat and spoil are biased towards small-sized target objects.



**Figure 13.** Statistical plot of the classes in the RiverInspect-2024 dataset.

The RiverInspect-2024 dataset encompasses both regularly shaped ship objects and irregularly shaped objects such as accumulated garbage, spoil, and pooled water objects. Recognizing irregularly shaped targets poses significant challenges due to variations in object colors, weather conditions during capture, and diverse backgrounds. Specific examples of UAV aerial photography effects and detection are shown in Figure 14.

Using RiverInspect-2024 as the dataset for the generalization test, the final comparison results of each model are shown in Table 5.



**Figure 14.** Examples of RiverInspect-2024. It is used for a river drone-inspection project.

**Table 5.** Experiment results of models on the RiverInspect-2024 dataset (The bold denotes the best effect and highlights the algorithm we proposed).

Models	Precision	Recall	mAP@50
SSD [49]	0.867	0.849	0.876
Faster R-CNN [50]	0.623	0.825	0.761
Cascade R-CNN [13]	0.67	0.771	0.741
YOLOv3 [51]	0.851	0.713	0.783
YOLOv5 [52]	0.855	<b>0.854</b>	0.892
YOLOv7 [53]	0.865	0.831	0.88
YOLOv8n [23]	0.834	0.845	0.896
RT-DETR [54]	0.855	0.843	0.899
YOLOv9 [55]	0.848	0.819	0.888
Ground DINO 1.5 [56]	0.839	0.832	0.881
YOLOv10 [57]	0.846	0.839	0.896
<b>YOLO-DroneMS</b>	<b>0.893</b>	0.843	<b>0.906</b>

The mAP@50 for YOLO-DroneMS is 0.906, which is the highest value in all models, which reflects the overall performance of the model in object detection tasks, showing that YOLO-DroneMS excels in accurately detecting objects with high confidence levels. At the same time, it is noted that compared with the advanced models (RT-DETR, YOLOv9, Ground DINO 1.5, and YOLOv10), YOLO-DroneMS still has certain advantages, indicating that it is highly competitive in UAV target detection. And the performance on the private dataset shows that YOLO-DroneMS has a certain robustness.

## 6. Conclusions and Discussion

This study explores the distribution of objects in the VisDrone dataset, focusing on UAV detection scenarios. On the Visdrone dataset, model improvement and optimization are conducted based on the YOLOv8 baseline. This optimization effectively increases the model's receptive field, enhancing its capability for UAV detection tasks. Firstly, this study proposes the LSKA mechanism to SPPF, and weighted processing of multi-scale feature

maps is performed, which allows the model to focus more on features relevant to the target, thereby enhancing detection accuracy. For the neck, ASF-DySample introduces attention scale sequence fusion and dynamic upsampling to enhance the fusion of high-dimensional information from deep feature maps with detailed information from shallow feature maps. Then, the C2f structure is optimized using iRMB-DRB, which balances the advantages of dynamic global modeling and static local information fusion, making it easier to capture fine details and contextual information from UAV images. Finally, replacing the original CIoU with WIoUv3 makes the YOLO-DroneMS focus on anchoring boxes of standard quality, dynamically optimizing the weighting for small objects to improve detection performance.

YOLO-DroneMS also achieves the best performance on the private dataset in generalization experiments.

UAV object detection faces several challenges and issues in practice, including: 1. Small target detection: Small objects may be difficult to accurately detect due to factors such as low resolution and low contrast. 2. Target occlusion and dense scenes: In dense scenes, occlusion between objects may occur, resulting in some targets being occluded by others. 3. Variations in lighting and weather conditions: UAVs capture images under different lighting and weather conditions, such as shadows, direct sunlight, or rainy and foggy weather, leading to changes in the appearance and features of objects. 4. Real-time requirements: Many applications require real-time UAV object detection, such as in monitoring and emergency response fields.

Future research will utilize multi-scale features that can improve the detection capability for targets of different sizes, considering that the relationship and contextual information between targets and their surroundings can enhance the detector's ability to handle occlusion and dense scenes and design lightweight model structures and algorithms to reduce the computational and storage costs of models to enable efficient target detection on resource-constrained UAV platforms.

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

**Funding:** This study was supported by the Research Project of China Water Resources Pearl River Planning Surveying & Designing Co., Ltd. (2023KY01, 2022KY06 and 2022KY04).

**Data Availability Statement:** The original contributions presented in the study are included in the article. Further inquiries can be directed to the corresponding author.

**Conflicts of Interest:** Author Xueqiang Zhao was employed by the company China Water Resources Pearl River Planning Surveying & Designing Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## References

1. Keawboontan, T.; Thammawichai, M. Towards Real-Time UAV Multi-Target Tracking using Joint Detection and Tracking. *IEEE Access* **2023**, *11*, 65238–65254. [[CrossRef](#)]
2. La Salandra, M.; Colacicco, R.; Dellino, P.; Capolongo, D. An Effective Approach for Automatic River Features Extraction Using High-Resolution UAV Imagery. *Drones* **2023**, *7*, 70. [[CrossRef](#)]
3. Xing, W.J.; Cui, Z.C.; Qi, J. HRCTNet: A hybrid network with high-resolution representation for object detection in UAV image. *Complex. Intell. Syst.* **2023**, *9*, 6437–6457. [[CrossRef](#)]
4. Ren, J.; Wang, Y. Overview of Object Detection Algorithms Using Convolutional Neural Networks. *J. Comput. Commun.* **2022**, *10*, 115–132. [[CrossRef](#)]
5. Zhang, Y.; Li, X.; Wang, F.; Wei, B.; Li, L. A Comprehensive Review of One-stage Networks for Object Detection. In Proceedings of the 2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), Xi'an, China, 17–20 August 2021; pp. 1–6.
6. Babulal, K.S.; Das, A.K. *Deep Learning-Based Object Detection: An Investigation*; Springer Nature: Singapore, 2022; pp. 697–711.
7. Du, L.; Zhang, R.; Wang, X. Overview of two-stage object detection algorithms. *J. Phys.* **2020**, *1544*, 12033. [[CrossRef](#)]

8. Du, D.; Zhu, P.; Wen, L.; Bian, X.; Ling, H.; Hu, Q.; Peng, T.; Zheng, J.; Wang, X.; Zhang, Y.; et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 213–226.
9. Du, D.; Qi, Y.; Yu, H.; Yang, Y.; Duan, K.; Li, G.; Zhang, W.; Huang, Q.; Tian, Q. The unmanned aerial vehicle benchmark object detection and tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 370–386.
10. Freudenmann, L.; Sommer, L.; Schumann, A. Exploitation of data augmentation strategies for improved uav detection. In *Automatic Target Recognition XXXI*; SPIE: Bellingham, WA, USA, 2021; pp. 119–132.
11. Sun, Z.; Dai, M.; Leng, X.; Lei, Y.; Xiong, B.; Ji, K.; Kuang, G. An anchor-free detection method for ship targets in high-resolution SAR images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 7799–7816. [[CrossRef](#)]
12. Zand, M.; Etemad, A.; Greenspan, M. Objectbox: From centers to boxes for anchor-free object detection. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; pp. 390–406.
13. Cai, Z.; Vasconcelos, N. Cascade R-CNN: High quality object detection and instance segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1483–1498. [[CrossRef](#)]
14. Yang, F.; Fan, H.; Chu, P.; Blasch, E.; Ling, H. Clustered object detection in aerial images. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8311–8320.
15. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. Centernet: Keypoint triplets for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6569–6578.
16. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
17. Tan, M.; Pang, R.; Le, Q.V. Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10781–10790.
18. 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.
19. Merenda, M.; Porcaro, C.; Iero, D. Edge machine learning for ai-enabled iot devices: A review. *Sensors* **2020**, *20*, 2533. [[CrossRef](#)]
20. Zhang, H.; Li, F.; Liu, S.; Zhang, L.; Su, H.; Zhu, J.; Ni, L.M.; Shum, H.Y. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv* **2022**, arXiv:2203.03605.
21. Chen, K.; Liu, C.; Chen, H.; Zhang, H.; Li, W.; Zou, Z.; Shi, Z. RSPrompter: Learning to prompt for remote sensing instance segmentation based on visual foundation model. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 1–17. [[CrossRef](#)]
22. Soylu, E.; Soylu, T. A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition. *Multimed. Tools Appl.* **2024**, *83*, 25005–25035. [[CrossRef](#)]
23. Sohan, M.; Sai Ram, T.; Reddy, R.; Venkata, C. A Review on YOLOv8 and Its Advancements. In Proceedings of the International Conference on Data Intelligence and Cognitive Informatics, Tirunelveli, India, 18–20 November 2024; pp. 529–545.
24. Vijayakumar, A.; Vairavasundaram, S. YOLO-based Object Detection Models: A Review and its Applications. *Multimed. Tools Appl.* **2024**, *83*, 83535–83574. [[CrossRef](#)]
25. Hussain, M. YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection. *Machines* **2023**, *11*, 677. [[CrossRef](#)]
26. Tang, X.; Ru, X.; Su, J.; Adonis, G. A Transmission and Transformation Fault Detection Algorithm Based on Improved YOLOv5. *Comput. Mater. Contin.* **2023**, *76*, 2997–3011. [[CrossRef](#)]
27. 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](#)]
28. Xue, Z.; Lin, H.; Wang, F. A small target forest fire detection model based on YOLOv5 improvement. *Forests* **2022**, *13*, 1332. [[CrossRef](#)]
29. Yu, H.; Li, X.; Feng, Y.; Han, S. Multiple attentional path aggregation network for marine object detection. *Appl. Intell.* **2023**, *53*, 2434–2451. [[CrossRef](#)]
30. Wan, G.; Fang, H.; Wang, D.; Yan, J.; Xie, B. Ceramic tile surface defect detection based on deep learning. *Ceram. Int.* **2022**, *48*, 11085–11093. [[CrossRef](#)]
31. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. Yolox: Exceeding yolo series in 2021. *arXiv* **2021**, arXiv:2107.08430.
32. Zhang, Y.-F.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* **2022**, *506*, 146–157. [[CrossRef](#)]
33. Du, F.-J.; Jiao, S.-J. Improvement of lightweight convolutional neural network model based on YOLO algorithm and its research in pavement defect detection. *Sensors* **2022**, *22*, 3537. [[CrossRef](#)] [[PubMed](#)]
34. Lau, K.W.; Po, L.-M.; Rehman, Y.A.U. Large separable kernel attention: Rethinking the large kernel attention design in cnn. *Expert Syst. Appl.* **2024**, *236*, 121352. [[CrossRef](#)]
35. Wang, Z.; Li, Y.; Liu, Y.; Meng, F. Improved object detection via large kernel attention. *Expert Syst. Appl.* **2024**, *240*, 122507. [[CrossRef](#)]
36. Wang, J.; Chen, K.; Xu, R.; Liu, Z.; Loy, C.C.; Lin, D. Carafe: Content-aware reassembly of features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3007–3016.

37. Kang, M.; Ting, C.-M.; Ting, F.F.; Phan, R.C.-W. ASF-YOLO: A novel YOLO model with attentional scale sequence fusion for cell instance segmentation. *Image Vision Comput.* **2024**, *147*, 105057. [[CrossRef](#)]
38. Liu, W.; Lu, H.; Fu, H.; Cao, Z. Learning to Upsample by Learning to Sample. In *Proceedings of the Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023*; pp. 6027–6037.
39. Shu, X.; Zhang, L. Research on PointPillars Algorithm Based on Feature-Enhanced Backbone Network. *Electronics* **2024**, *13*, 1233. [[CrossRef](#)]
40. Zhang, J.; Li, X.; Li, J.; Liu, L.; Xue, Z.; Zhang, B.; Jiang, Z.; Huang, T.; Wang, Y.; Wang, C. Rethinking mobile block for efficient attention-based models. In *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023*; pp. 1389–1400.
41. Chiang, H.-Y.; Frumkin, N.; Liang, F.; Marculescu, D. MobileTL: On-device transfer learning with inverted residual blocks. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023*; pp. 7166–7174.
42. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [[CrossRef](#)]
43. Ding, X.; Zhang, Y.; Ge, Y.; Zhao, S.; Song, L.; Yue, X.; Shan, Y. Unireplknet: A universal perception large-kernel convnet for audio, video, point cloud, time-series and image recognition. *arXiv* **2023**, arXiv:2311.15599.
44. Tong, Z.; Chen, Y.; Xu, Z.; Yu, R. Wise-IOU: Bounding box regression loss with dynamic focusing mechanism. *arXiv* **2023**, arXiv:2301.10051.
45. Liu, C.; Meng, F.; Zhu, Z.; Zhou, L. Object Detection of UAV Aerial Image based on YOLOv8. *Front. Comput. Intell. Syst.* **2023**, *5*, 46–50. [[CrossRef](#)]
46. Zhang, L.; He, Y.; Zhou, Y.; Chen, Y.; Chen, Z.; Yang, Y. An improved YOLOv7 algorithm for laser welding defect detection. In *Proceedings of the Sixth International Conference on Computer Information Science and Application Technology (CISAT 2023), Hangzhou, China, 26–28 May 2023*; pp. 343–349.
47. Powers, D.M. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061.
48. Revaud, J.; Almazán, J.; Rezende, R.S.; Souza, C.R.d. Learning with average precision: Training image retrieval with a listwise loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019*; pp. 5107–5116.
49. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Proceedings, Part. I 14. Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.
50. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]
51. Farhadi, A.; Redmon, J. Yolov3: An incremental improvement. In *Computer Vision and Pattern Recognition*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 1804, pp. 1–6.
52. Jocher, G.; Chaurasia, A.; Stoken, A.; Borovec, J.; Kwon, Y.; Fang, J.; Michael, K.; Montes, D.; Nadar, J.; Skalski, P.; et al. Ultralytics/yolov5: v6. 1-tensorrt, tensorflow edge tpu and opencv export and inference. *Zenodo* **2022**. [[CrossRef](#)]
53. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.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.
54. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. Detsr beat yolos on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 17–21 June 2024*; pp. 16965–16974.
55. Wang, C.Y.; Yeh, I.H.; Liao, H.Y.M. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. *arXiv* **2024**, arXiv:2402.13616.
56. Ren, T.; Jiang, Q.; Liu, S.; Zeng, Z.; Liu, W.; Gao, H.; Huang, H.; Ma, Z.; Jiang, X.; Chen, Y.; et al. Grounding DINO 1.5: Advance the “Edge” of Open-Set Object Detection. *arXiv* **2024**, arXiv:2405.10300.
57. 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.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.