sensors

MDPI

Article

YOLOv4 with Deformable-Embedding-Transformer Feature Extractor for Exact Object Detection in Aerial Imagery

Yiheng Wu and Jianjun Li *

College of Computer and Information Engineering, Central South University of Forestry and Technology University, Changsha 410004, China
* Correspondence: t20010539@csuft.edu.cn; Tel.: +86-137-5513-6109

Abstract: The deep learning method for natural-image object detection tasks has made tremendous progress in recent decades. However, due to multiscale targets, complex backgrounds, and high-scale small targets, methods from the field of natural images frequently fail to produce satisfactory results when applied to aerial images. To address these problems, we proposed the DET-YOLO enhancement based on YOLOv4. Initially, we employed a vision transformer to acquire highly effective global information extraction capabilities. In the transformer, we proposed deformable embedding instead of linear embedding and a full convolution feedforward network (FCFN) instead of a feedforward network in order to reduce the feature loss caused by cutting in the embedding process and improve the spatial feature extraction capability. Second, for improved multiscale feature fusion in the neck, we employed a depth direction separable deformable pyramid module (DSDP) rather than a feature pyramid network. Experiments on the DOTA, RSOD, and UCAS-AOD datasets demonstrated that our method's average accuracy (mAP) values reached 0.728, 0.952, and 0.945, respectively, which were comparable to the existing state-of-the-art methods.

Keywords: aerial imagery; ultra-high spatial resolution orbital imagery; object detection; YOLOv4; vision transformer; deep learning

Citation: Wu, Y.; Li, J. YOLOv4 with Deformable-Embedding-Transformer Feature Extractor for Exact Object Detection in Aerial Imagery. *Sensors* **2023**, *23*, 2522. https://doi.org/10.3390/s23052522

Academic Editors: Andreas Savakis, Erik Blasch and Kannappan Palaniappan

Received: 21 November 2022
Revised: 21 February 2023
Accepted: 23 February 2023
Published: 24 February 2023

1. Introduction

Object detection in ultra-high spatial resolution aerial images aims to rapidly determine a target's location and classify the target. Object detection in aerial images is widely used in numerous important fields, including the localization and identification of military targets [1], natural environment protection [2], disaster detection [3], and urban construction planning [4].

The application of convolutional neural networks to computer vision and object detection has made significant strides in recent years. Successively, superior object detection algorithms, including the R-CNN series [5–8], the YOLO series [9–11], and the SSD series [12–14], have been proposed. With the introduction of ViT [15], the transformer is widely utilized in the field of image processing. Recently, transformer-based detectors such as DETR [16] and DINO [17] have been proposed and have performed better in some detection tasks than CNN-based networks. With large annotated natural image datasets such as MS COCO [18] and PASCAL VOC [19], these CNN-based and transformer-based target detection algorithms have achieved astonishing results. However, the above models have poor detection performance with aerial images, which is due to the specificity of aerial images [20]. Firstly, aerial images are taken at altitudes ranging from several hundred meters to tens of kilometers due to differences in observation equipment, which results in diverse sizes of targets in the same category. Secondly, aerial images are usually taken by acquisition equipment from a top view. As a result, the background of aerial images is complex and contains a great deal of redundant information, causing severe interference in detection. Finally, since aerial images are captured from high altitudes, they contain

an abundance of small targets. For instance, a car's pixels might be 5×5 or even smaller. This makes it challenging to extract edge features from aerial images of targets and to distinguish targets from the background. These characteristics make aerial image object detection a unique and difficult problem.

In this research, we focus on addressing the issues mentioned above to enhance the target detection performance in aerial images. In addition, the speed of detection is a significant obstacle for detection algorithms, as target detection from satellite imagery is typically performed in real time. A one-stage object detection algorithm, exemplified by the YOLO series [9–11], combines object classification and localization into a single-stage regression problem, vastly enhancing the detection speed in comparison to a traditional two-stage object detection algorithm of localization followed by classification. Currently, the most advanced versions of YOLO are YOLOv4 [21], YOLOv5, and YOLOX [22]. Compared to YOLOv4, YOLOv5 and YOLOX utilize a large number of dense connection structures; their models are more complex, and their detection speed is decreased, but their detection accuracy is comparable. Therefore, we enhanced the feature extraction network and the feature fusion network based on YOLOv4 to meet the detection requirements for targets in optical images with ultra-high spatial resolutions, particularly small targets.

DET-YOLO is the name of the enhanced YOLOv4 variant. Experimental results showed that our proposed DET-YOLO outperformed the general YOLOv4 in detecting targets in aerial images.

This study's contribution is summarized as follows:

1. A new feature extraction network named a deformable embedding vision transformer (DEViT) is proposed, which has the excellent global feature extraction capability of transformers while using deformable embedding instead of normal embedding to efficiently extract features at different scales. In addition, a fully convolutional feedforward network (FCFN) is used to replace the feedforward neural network (FFN), and the extraction of location information is effectively enhanced by introducing zero padding and convolutional operations.
2. The depthwise separable deformable convolution (DSDC) is proposed to reduce computational effort while preserving the deformable convolution's ability to zero in on regions of interest. It is proposed that the depthwise separable deformable pyramid (DSDP) module extracts multiscale feature maps and prioritizes key features.
3. Our proposed DET-YOLO achieved the highest accuracy among existing models, with mean average precision (mAP) values of 0.728 on the DOTA dataset, 0.952 on the RSOD dataset, and 0.945 on the UCAS-AOD dataset.

The rest of the paper is organized as follows: Section 2 describes the proposed network and the individual components; Section 3 presents the dataset and the evaluation metrics used; Section 4 gives the experimental results and analyses to demonstrate the accuracy and validity of the proposed algorithm; and Section 5 gives the conclusions.

2. Methodology

2.1. Review of YOLOv4

YOLOv4's architecture comprises three primary components: the backbone, the neck, and the predicting head. The backbone extracts feature data from an image input. The neck collects and combines multiscale feature data to generate three distinct-scale feature maps. Based on these created feature maps, the predicting head detects objects. YOLOv4 employs the CSPDarknet53 [23] framework as its backbone, SPP [24] and PANet [25] as the neck, and YOLO's detection head. In addition, YOLOv4 uses a number of measures to enhance the performance of target detection, including mosaic data enhancement, Mish activation functions, and CIOU loss.

2.2. DET-YOLO

Figure 1 depicts the architecture of our proposed DET-YOLO for object detection in aerial images. We employ DEViT as the backbone, DSDP and PANet as the neck, and

YOLO's detection head as the predicting head. In addition, it employs the Mish [26] activation function in place of the leaky ReLU [27] activation function, CIoU [28] loss in place of the ordinary IoU [29] loss, and the cosine annealing strategy [30] for learning rate attenuation to achieve more accurate detection results. In the subsequent subsections, we elaborate on DEViT and DSDP principles.

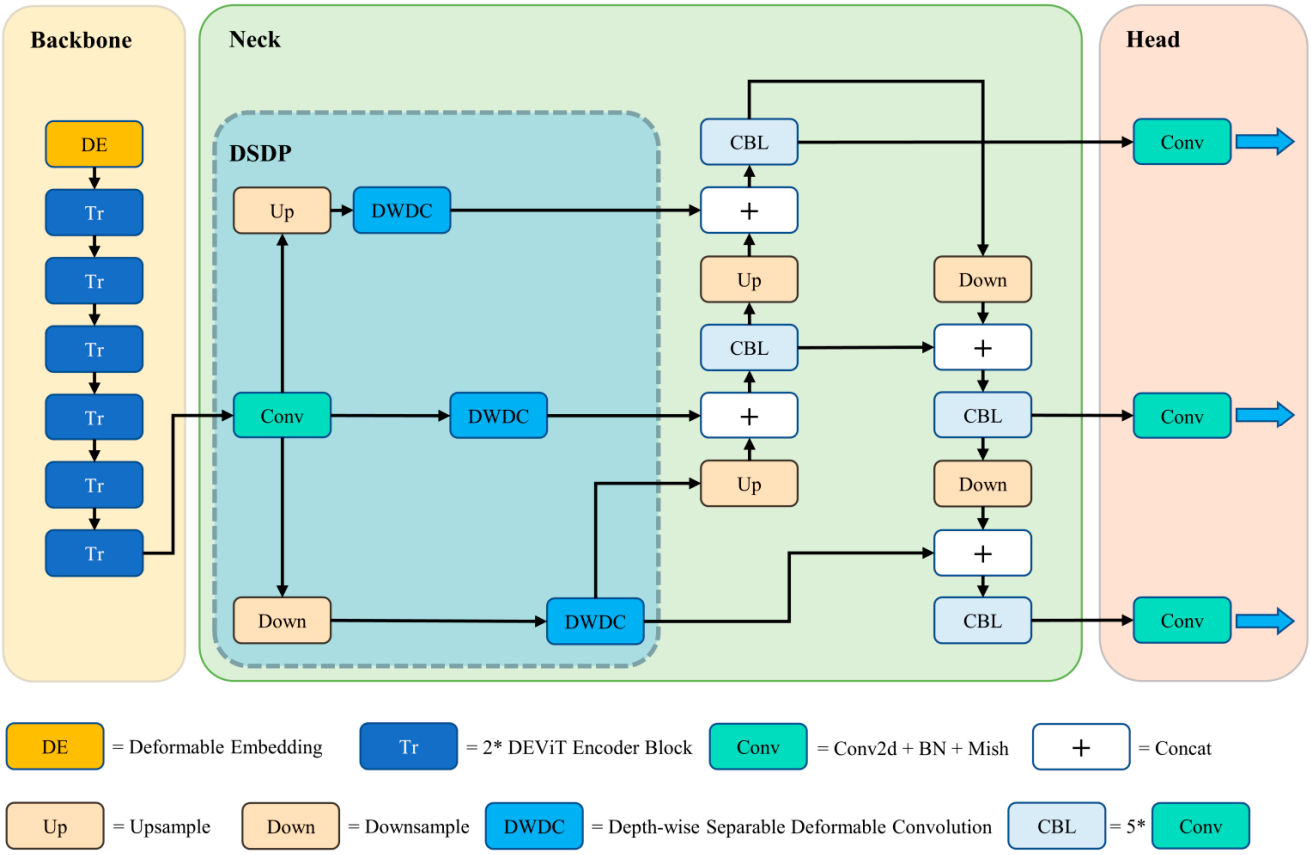


Figure 1. An overview of DET-YOLO. We use DEViT as the backbone, as opposed to CSPDarknet53 in the original implementation. In addition, the DSDP module is utilized in place of skip connections and SPP in order to obtain multiscale feature maps.

2.3. Deformable Embedding Vision Transformer

Figure 2 depicts the structure of a non-hierarchical adaptive transformer, DEViT, that is proposed to have an excellent global feature capture capability while efficiently and flexibly extracting target information. We made the following enhancements: First, we proposed deformable embedding to capture targets of varying sizes in aerial images. Second, we proposed fully convolutional feedforward networks to improve the location information extraction.

These two aspects are elaborated upon below.

2.3.1. Deformable Embedding

Traditional visual transformers, including hierarchical models such as PVT [31] and the Swin Transformer [32] as well as non-hierarchical models such as ViT [15] and DEiT [33], use a fixed-size patch embedding based on the implicit assumption that the fixed image split design is appropriate for all images. Given the input image $X \in \mathbb{R}^{C \times H \times W}$, where C, H, and W represent the channel dimension, height, and width of the feature, respectively, a vanilla patch embedding splits X into N patches of size $P \times P$ ($N = HW/P^2$). The patches are then connected to form a sequence of flattened 2D patches ($X_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$), each of which represents a rectangular region of input image X.

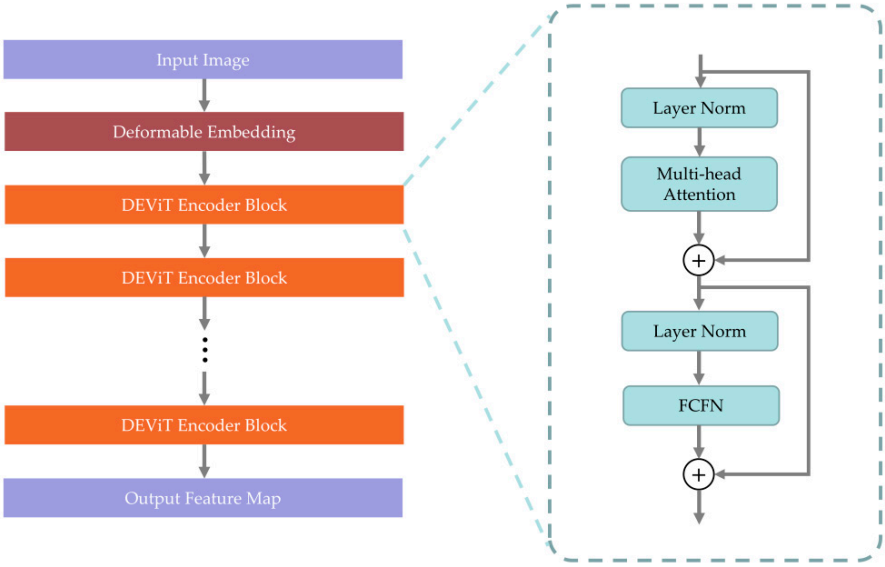


Figure 2. The DEViT architecture.

Nevertheless, the scale of the information contained in aerial images varies greatly due to the different observation carriers and acquisition methods, and the typical embedding methods frequently do not fit well or even destroy the target feature information, as depicted in Figure 3. Therefore, a data-dependent sparse embedding strategy is required for the flexible construction of relevant features.

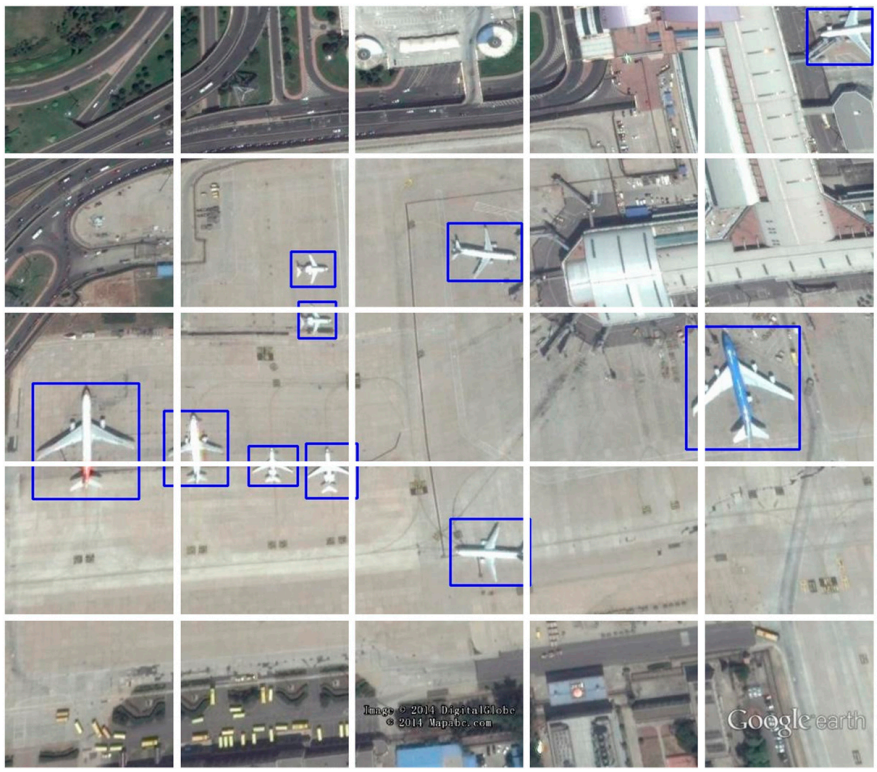
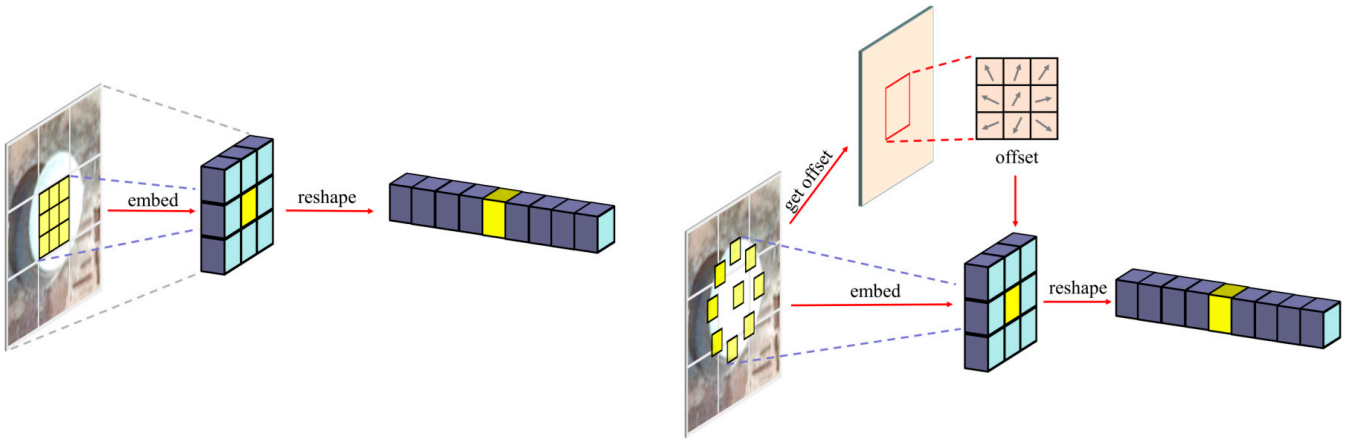


Figure 3. Vanilla patch embedding example. Patch splitting is represented by the white lines, while the ground-truth boxes are represented by the blue lines. Some targets' semantics are compromised.

DCN [34] proposes a deformable convolution, which allows the convolution kernel to be adaptively focused on the target region by obtaining an offset, as opposed to being limited to a fixed size and shape of the perceptual field. In response, we propose deformable

embedding, which obtains the ability to extract objects of different scales by adaptively acquiring an offset, as depicted in Figure 4b.



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

Figure 4. Comparison of vanilla patch embedding and deformable embedding: (**a**) vanilla patch embedding and (**b**) deformable embedding.

As depicted in Figure 4a, the above embedding process can also be viewed as downsampling X using a linear embedding layer with kernel size P and strike size P to obtain the feature map $X' \in \mathbb{R}^{(P^2 \cdot C) \times \frac{H}{P} \times \frac{W}{P}}$ and then reshaping the feature map X' into X_p. The process of embedding can be described as follows:

$$X' = Embed(X), \tag{1}$$

$$X_p = Reshape(X'), \tag{2}$$

where $Embed(\cdot)$ is the operation for linear embedding and $Reshape(\cdot)$ is the flattening function.

For each patch $p(i)$ on feature map X', a rectangular region of the input image with the area $P \times P$ is represented. The coordinates of its lower left corner are denoted as $l_i^1 = (x_i^1, y_i^1)$, and the sequence of coordinates corresponding to $p(i)$ is denoted as $\left\{l_i^1, \cdots, l_i^{P \times P}\right\}$. These locations' features are represented as $\left\{f_i^{\,j}, \cdots, f_i^{\,P \times P}\right\}$. Equation (3) illustrates how linear embedding flattens these features and processes them with a linear layer to produce a representation of the patch.

$$p(i) = W_{patch} \cdot concat\left(f_i^{\,j}, \cdots, f_i^{\,P \times P}\right) + b_{patch}. \tag{3}$$

Feature map X is fed to the lightweight subnetwork $\theta_{offset}(\cdot)$, which has a single convolutional layer with a 1×1 convolution kernel. The bias quantity feature map $X_{offset} \in \mathbb{R}^{2C \times H \times W}$ is produced, and the layer $2i-1$ and layer $2i$ channels represent the x- and y-axis offsets, respectively, for to the layer i channel of the original map. The offset associated with each position l_i^j is $\Delta l_i^j = \theta_{offset}(l_i^j)$. Thus, Equation (4) depicts the representation of the new patch obtained via deformable embedding.

$$p(i) = W_{patch} \cdot concat\left(f_i^{\,j}\left(l_i^1 + \Delta l_i^1\right), \cdots, f_i^{\,P \times P}\left(l_i^{P \times P} + \Delta l_i^{P \times P}\right)\right) + b_{patch}, \tag{4}$$

where $p(i)$ constitutes the new feature graph X'. X' is substituted into Equation (2) to obtain the deformable embedding result X_p. The procedure for $p\,(i)$ is as follows:

$$X_p = concat(p(1), \cdots, p(N)). \tag{5}$$

Our proposed deformable embedding receives a 2D offset on each pixel for each patch, allowing it to extract the target information adaptively and effectively avoiding the issue of semantic information corruption during vanilla splitting.

2.3.2. Full Convolution Feedforward Network

A vanilla feedforward neural network (FFN) of visual transformers consists of two layers of MLP composition and a GELU [35] activation function, as demonstrated by Equation (6).

$$\mathrm{FFN}(F) = \mathrm{MLP}(\mathrm{GELU}(\mathrm{MLP}(F))). \tag{6}$$

Due to the high computational cost of MLPs, they require a substantial amount of computation while effectively preserving the dimensionality of the feature channels. As shown in Figure 5, we propose an FCFN with enhancements to both reduce the computational effort and enhance the capability of feature extraction.

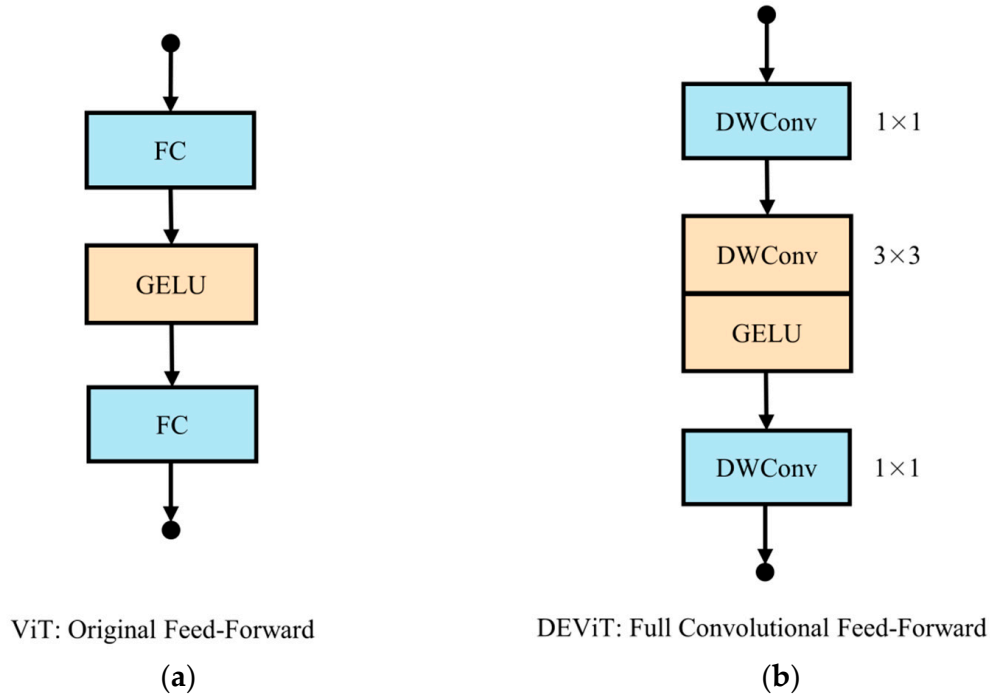


Figure 5. Comparison of FFN and FCFN: (**a**) the vanilla FFN of a vision transformer; (**b**) the FCFN of DEViT.

First, instead of MLP, we use two 1×1 depthwise separable convolutions [36] to improve the model's robustness while further reducing the computational effort. Second, we enhance the network's capacity to extract location data. The study by Islam et al. [37] demonstrated that the use of convolutional layers with zero padding can aid a network in learning absolute position data. Before the GELU activation function, we introduce a depth separable convolution with a stride of 1, a kernel size of 3, and a padding size of 1, which only marginally increases the computational effort. As shown in Equation (7), the combination of these two enhancements forms our fully convolutional neural network.

$$\mathrm{FCFN}(F) = (\mathrm{DWConv1}_{1\times1}(\mathrm{GELU}(\mathrm{DWConv}_{3\times3}(\mathrm{DWConv1}_{1\times1}(F))))). \tag{7}$$

To visually demonstrate the improvement in the computational requirements of our FCFN over the FFN, we compare their computational costs and parameter counts.

A standard MLP layer accepts a $1 \times M_{mlp}$ feature sequence (I) as an input and outputs a $1 \times N_{mlp}$ feature sequence (O), where M_{mlp} and N_{mlp} represent the lengths of the input

and output feature sequences, respectively. MLP contains N_{mlp} neurons (n). The standard MLP output feature sequence is computed as follows:

$$O_n = \sum_{i,j} n_i \cdot I_j. \tag{8}$$

The standard MLP incurs the following computational cost:

$$cost_{mlp} = M_{mlp} \cdot N_{mlp}. \tag{9}$$

The standard MLP parameters are as follows:

$$param_{mlp} = M_{mlp} \cdot (N_{mlp} + 1). \tag{10}$$

For FFN, the first MLP receives a $1 \times (P^2 \cdot C)$ feature sequence as an input and outputs a $1 \times (P^2 \cdot 4C)$ feature sequence, while the second MLP receives a $1 \times (P^2 \cdot 4C)$ feature sequence as an input and outputs a $(P^2 \cdot C)$ feature sequence. Therefore, the computational and parametric quantities of the FFN are shown in Equations (11) and (12), respectively.

$$\begin{aligned} Cost_{ffn} &= (P^2 \cdot C) \cdot (P^2 \cdot 4C) + (P^2 \cdot 4C) \cdot (P^2 \cdot C) \\ &= 8 \cdot P^4 \cdot C^2, \end{aligned} \tag{11}$$

$$\begin{aligned} param_{ffn} &= (P^2 \cdot C) \cdot (P^2 \cdot 4C + 1) + (P^2 \cdot 4C) \cdot (P^2 \cdot C + 1) \\ &= 8 \cdot P^4 \cdot C^2 + 5 \cdot P^2 \cdot C. \end{aligned} \tag{12}$$

A standard convolutional layer receives an $M_{conv} \times W \times H$ feature map (D) as an input and generates an $N_{conv} \times A \times B$ feature map (G) as an output, where W and H represent the width and height of the input feature map, M_{conv} represents the number of input channels, A and B represent the width and height of the output feature map, and N_{conv} represents the number of output channels. The convolution layer contains N_{conv} convolution kernels (K^n) with an $M_{conv} \times w \times h$ shape and a strike size of 1. The output feature sequence of a convolutional layer is computed as follows:

$$G_{k,l,n} = \sum_{i,j,m} K^n_{i,j,m} \cdot D_{k+i-1,l+j-1,m}. \tag{13}$$

A standard convolutional layer incurs the following computational expenses:

$$cost_{conv} = w \cdot h \cdot M_{conv} \cdot N_{conv} \cdot A \cdot B. \tag{14}$$

The parameters of a standard convolutional layer are:

$$param_{conv} = (w \cdot h \cdot N_{conv} + 1) \cdot M_{conv}. \tag{15}$$

The computational cost of the depthwise separable convolution corresponding to the standard convolution is:

$$cost_{dwconv} = w \cdot h \cdot M_{conv} \cdot A \cdot B + M_{conv} \cdot N_{conv} \cdot A \cdot B. \tag{16}$$

The computational cost of the depthwise separable convolution corresponding to the standard convolution is:

$$param_{dwconv} = (w \cdot h + 1) \cdot M_{conv} + (M_{conv} + 1) \cdot N_{conv}. \tag{17}$$

For the FCFN, the first DWConv takes a $C \times P \times P$ feature map as an input and returns a $4C \times P \times P$ feature map; the second DWConv takes a $4C \times P \times P$ feature map as an input and returns a $4C \times P \times P$ feature map; and the third DWConv takes a $4C \times P \times P$ feature

map as an input and returns a $C \times P \times P$ feature map. Therefore, Equations (18) and (19) display the computational and parametric quantities of the FCFN.

$$\begin{aligned} cost_{fcfn} = \ & (1{\cdot}C{\cdot}P^{\,2}{+}4C{\cdot}C{\cdot}P^{\,2}) + (9{\cdot}4C{\cdot}P^{\,2}{+}4C{\cdot}4C{\cdot}P^{\,2}) + (1{\cdot}C{\cdot}P^{\,2}{+}4C{\cdot}C{\cdot}P^{\,2}) \\ & = 24{\cdot}P^{\,2}{\cdot}C^{\,2} + 41{\cdot}P^{\,2}{\cdot}C, \end{aligned} \tag{18}$$

$$\begin{aligned} param_{fcfn} = \ & (1{\cdot}C{\cdot}P^{\,2}{+}C + 4C{\cdot}C) + (9{\cdot}C + 4C{\cdot}4C{\cdot}P^{\,2}) + (1{\cdot}C{\cdot}P^{\,2}{+}4C{\cdot}C{\cdot}P^{\,2}) \\ & = 24{\cdot}C^{\,2}{+}17{\cdot}C. \end{aligned} \tag{19}$$

By substituting the FFN with the FCFN, the following reductions in computation and parameters are obtained:

$$\frac{cost_{fcfn}}{cost_{ffn}} = \frac{41}{8P^{\,2}C} + \frac{3}{P^{\,2}}, \tag{20}$$

$$\frac{param_{fcfn}}{param_{ffn}} = \frac{17 + 24C}{8P^{\,4}C{+}5P^{\,2}}. \tag{21}$$

DEViT employs a deformable embedding with a patch size of 16 and accepts high-resolution RGB satellite images as inputs; consequently, each FCFN is $\frac{113}{6144}$ of the computation and $\frac{89}{1574144}$ of the parameters of the corresponding standard FFN. As demonstrated in Section 4, the use of the FCFN rather than the FFN improves precision while significantly reducing the computational costs.

2.4. Depthwise Separable Deformable Pyramid

Instead of FPN, we propose a depthwise separable deformable pyramid to obtain feature maps at various scales from a non-hierarchical network architecture such as DEViT. The following subsections are descriptions of the DSDC and the DSDP, which comprises the DSDC.

2.4.1. Depthwise Separable Deformable Convolution

DSDC is proposed by combining depthwise separable convolution [36] and deformable convolution [34], which significantly reduces the computational effort while adaptively focusing on the object region, as is shown in Figure 6.

By adding an offset convolution layer, the vanilla deformable convolution layer learns offsets from the previous feature mapping. Using a convolution kernel of the same size as the corresponding convolution and an offset feature map with an output space the same size as the input feature map with twice as many channels as the original feature map, the offset of the convolution kernel relative to the x and y axes on each pixel of each channel is determined using the offset convolution.

Thus, the deformation can concentrate on objects in a dense, adaptive, and local manner. However, the additional convolution operation requires a substantial amount of computation, which slows model inference and makes it difficult to meet the real-time requirement for aerial image object detection. Deformable convolution yields a spatially two-dimensional offset that is channel-independent and two-dimensional. Therefore, we can separate the calculation of the offset from the calculation of the channel, which drastically reduces the amount of computation required without affecting the effect.

A standard convolution is decomposed into a depthwise convolution and a pointwise convolution by depthwise separable convolution. For each input channel, the depthwise convolution performs a separate convolution operation. The pointwise convolution then applies a 1×1 convolution to combine the outputs of the depthwise convolution. A depthwise separable convolution reduces the computational effort by decomposing a standard convolution into a channel-by-channel convolution kernel and a point-by-point convolution.

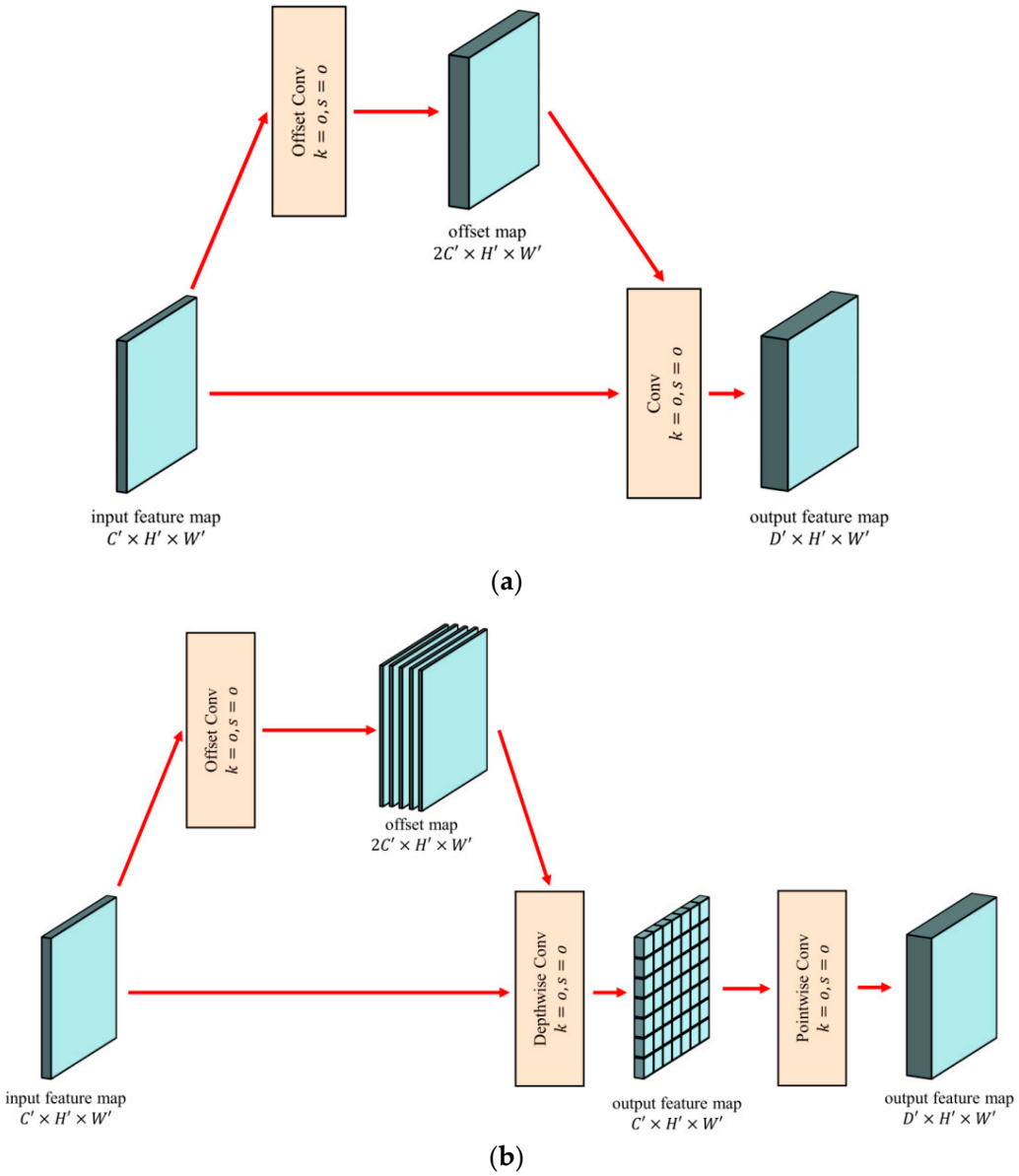


Figure 6. Comparison of vanilla deformable convolution and depthwise separable deformable convolution: (**a**) vanilla deformable convolution; (**b**) depthwise separable deformable convolution.

A standard deformable convolution is decomposed by DSDC into a deformable depthwise convolution and a pointwise convolution. A deformable depthwise convolution learns the offset independently for each channel, whereas a pointwise convolution combines the output of the deformable depthwise convolution for each channel. DSDC replaces the depthwise convolution portion of the depthwise separable convolution with a deformable convolution to gain the ability to adaptively focus on the region of interest. DSDC strikes a balance between improved computational efficiency and the deformable convolution's ability to extract features.

2.4.2. Depthwise Separable Deformable Pyramid Module

Because DEViT employs a non-hierarchical transformer structure, it is only possible to obtain a single-scale feature map. Due to the varying scales of the targets, a multiscale feature acquisition capability is required in aerial images. The most prevalent method for constructing a multiscale feature map is a feature pyramid network, which employs a top-down architecture with skip connections to combine features at various levels. For non-hierarchical transformers, Li et al. [38] attempted to use a simple feature pyramid

instead of a complex FPN structure, which effectively optimized the structure while maintaining accuracy.

We propose a simple multiscale feature acquisition module, dubbed a depthwise separable deformable pyramid, which only performs simple upsampling, downsampling, and a 1×1 convolution on the last layer of DEViT to generate different-scale feature maps. To focus the network on the region of interest and improve its ability to extract features, we introduce DSDC for each scale of features, causing the network to generate adaptive offsets for the target at each scale. Figure 7 depicts a comparison between DSDP and FPN.

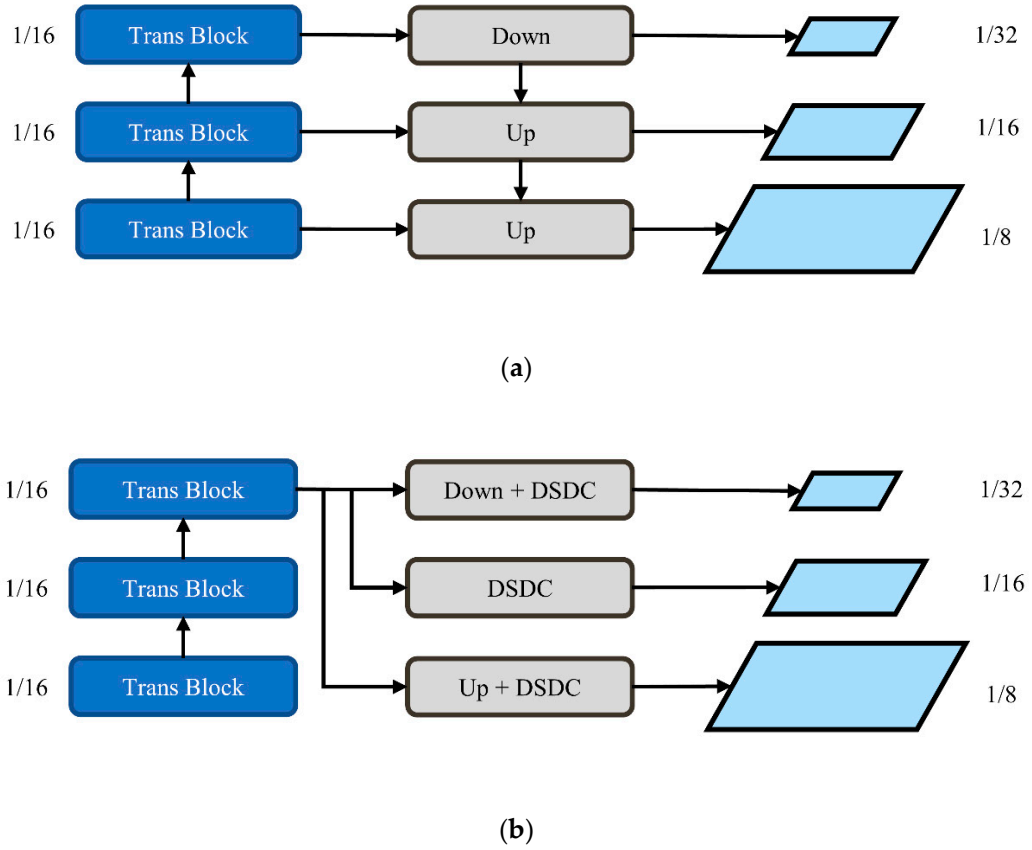


Figure 7. Comparison of feature pyramid network and depthwise separable deformable pyramid module: (**a**) FPN; (**b**) DSDC.

Subsequent experiments revealed that DSDP performed better than FPN on DET-YOLOv4, which will be discussed in detail in the ablation experiments subsection in Section 4.

3. Datasets and Experimental Settings

3.1. Data Description

To validate the efficacy of DET-YOLO, we conducted experiments on the widely used public datasets RSOD [39], UCAS-AOD [40], and DOTA [41] for multiclass object detection in orbital images with high spatial resolutions.

The RSOD dataset includes 6950 annotated samples and 976 high-resolution RGB satellite images from Google Earth and Tianditu. The dataset includes four distinct types of objects, including an oil tank, an aircraft, an overpass, and a playground.

The UCAS-AOD dataset is a high-resolution aerial image dataset designed to detect small and medium-sized targets. It consists of 1510 images and 14,596 annotated instances with a fixed image size of 1280×659. The dataset is divided into two categories, aircraft and vehicles, and contains counterexamples with no instances of objects.

The DOTA-v1.5 dataset was derived from Google Earth, the JL-1 satellite, and the GF-2 satellite, among other platforms and sensors. It contains 2806 RGB images and 188,282 annotated instances with dimensions ranging from 800 × 800 to 4000 × 4000. It includes targets for 16 categories such as planes, ships, storage tanks, baseball diamonds, tennis courts, basketball courts, ground track fields, harbors, bridges, large vehicles, small vehicles, helicopters, roundabouts, soccer fields, swimming pools, and container cranes. Compared to version 1.0, DOTA-v1.5 adds a large number of objects smaller than 10 pixels, making object detection tasks more difficult.

To validate the generality of our model, we also conducted experiments on UAVDT [41]. Unlike the orbital-image-based dataset described above, UAVDT is a large-scale challenging benchmark dataset based on UAV images. It contains approximately 80,000 frames with annotated information. When used for object detection, the UAVDT dataset has three types of objects (cars, trucks, and buses) and contains 23,829 training images and 16,580 test images at a size of 1024 × 540. The UAV images have a higher spatial resolution compared to the orbital images. As shown in Table 1, objects smaller than 10 pixels are referred to as very small objects, objects larger than 10 pixels but smaller than 50 pixels are referred to as small objects, objects with a size between 50 and 300 pixels are referred to as medium objects, and objects larger than 300 pixels are referred to as large objects, according to the classification criteria of [42]. In the orbital image datasets, the DOTA dataset has the highest proportion of small objects and is the most difficult of the three datasets, as shown in Table 1. UCAS-AOD focuses on small to medium-sized objects, while RSOD is moderately challenging. UAVDT focuses on small objects. For input into the network, we maintained the original sizes of the RSOD dataset, UCAS-AOD, and UAVDT. We separated them into training and validation sets based on criteria outlined in [41,43,44]. For the DOTA dataset, the oversized images were cropped. Approximately 20,000 1024 × 1024 images were generated in total. There were 1414 training images and 939 validation images in the original DOTA dataset. We obtained 14,729 training images and 5066 test images after cropping. The breakdown of the dataset is detailed in Table 2.

Table 1. Comparison of instance size distributions of the RSOD, UCAS-AOD, and DOTA datasets.

Datasets	Below 10 Pixels	10–50 Pixels	50–300 Pixels	Above 300 Pixels
RSOD	0%	36%	60%	4%
UCAS-AOD	0%	32%	68%	0%
DOTA	16%	69%	14%	1%
UAVDT	1%	89%	10%	0%

3.2. Evaluation Metrics

As evaluation metrics, we chose precision (P), recall (R), average precision (AP), and mean average precision (mAP), which are frequently employed in target detection.

P and R are defined as:

$$\mathrm{P} = \frac{TP}{TP + FP}, \tag{22}$$

$$\mathrm{R} = \frac{TP}{TP + FN}. \tag{23}$$

where TP is true positive, FP is false positive, TN is true negative, and FN is false negative.

AP is defined as the area of the P-R curve created using precision and recall as follows:

$$\mathrm{AP} = \int_0^1 P(R)dR, \tag{24}$$

where $P(\cdot)$ represents the P-R curve function. Using AP, we can precisely determine the model's capture capability for various objects.

The mAP is the average of the APs for all categories and accurately reflects the model's overall predictive performance. The definition is as follows:

$$\text{mAP} = \frac{1}{K}\sum_{i=1}^{K} AP_i, \tag{25}$$

where K represents the number of categories and AP_i represents the mean precision of the i-th category.

Table 2. Division of the RSOD, UCAS-AOD, and DOTA datasets.

Datasets		Class	Image Number
RSOD	Training Set	Oil Tank	102
		Aircraft	270
		Overpass	140
		Playground	140
	Test Set	Oil Tank	63
		Aircraft	176
		Overpass	36
		Playground	49
UCAS-AOD	Training Set	Aircraft	600
		Car	310
	Test Set	Aircraft	400
		Car	200
DOTA	Training Set	-	14,729
	Test Set	-	5066
UAVDT	Training Set	-	23,829
	Test Set	-	16,580

Using the aforementioned indicators, we could evaluate our strategy effectively.

3.3. Implementation Details

Our proposed DET-YOLO was implemented in Windows 10 using the 1.8.1 version of the PyTorch framework. The experiments were conducted on a desktop computer equipped with an Intel i9-7920X processor and an NVIDIA RTX 2080 ti graphics processing unit with 11 GB of memory.

We proposed that DEViT serve as the backbone of DET-YOLO in place of CSPDarkNet53. As a pre-trained model for DEViT, we partially inherited the ViT model trained on ImageNet using the MAE method. According to [45], we used the SGD optimizer with an initial learning rate of 0.01 and a momentum of 0.937 for training. According to [46], DET-YOLO was trained for 50 epochs on the RSOD and UCAS-AOD datasets and 150 epochs on the DOTA dataset. According to the characteristics of the different datasets [47,48], the images in the DOTA dataset were input to the network at a size of 1024 × 1024, whereas the images in the RSOD and UCAS-AOD datasets were resized to 608 × 608 to improve the feature extraction ability of small targets while maintaining a small network scale. We kept the original dimensions of the images in the UAVDT dataset as an input to the model.

4. Experimental Results and Discussion

4.1. Contrasting Experiments

To determine the efficacy and universality of DET-YOLO, we compared it to the most representative methods on the DOTA, RSOD, and UCAS-AOD datasets. The experimental outcomes for three distinct datasets are shown in Tables 3–6. Table 3 demonstrates that the mAP on the DOTA dataset was 0.728, which was an improvement of 0.012 over the suboptimal model SPH-YOLOv5, demonstrating that our model has excellent detection

performance for small and very small targets. Table 4 demonstrates that our model achieved an mAP of 0.952 on RSOD, which was an improvement of 0.066 over the suboptimal model YOLOv5 and substantiated the efficacy of medium and large target detection. Table 5 demonstrates that our model achieved an mAP of up to 0.945 on UCAS-AOD, which was an improvement of 0.070 over the suboptimal model YOLOv5, demonstrating that our model has excellent detection performance for small and medium-sized targets, such as vehicles and airplanes. Table 6 demonstrates that our method achieved an mAP of up to 0.424 on the UAVDT dataset, which was an improvement of 0.005 over the suboptimal model YOLOv5, demonstrating that our method is applicable to UAV images. In conclusion, the detection performance of our proposed DET-YOLO for aerial targets of various sizes, particularly small targets, was superior to that of the current principal methods, demonstrating the efficacy of this method.

Table 3. Comparison of the performances on the DOTA dataset.

Methods	*p*	R	mAP (IOU = 0.5)
Faster R-CNN	0.710	0.594	0.631
SSD	0.696	0.522	0.587
RetinaNet	0.714	0.585	0.622
YOLOv3	0.716	0.532	0.575
YOLOv4	0.732	0.593	0.653
YOLOv5	0.742	0.607	0.659
TPH-YOLOv5 [49]	0.785	0.643	0.683
SPH-YOLOv5 [47]	**0.806** *	**0.683**	0.716
DET-YOLO	0.748	0.668	**0.728**

* Bold indicates the best result in the current table. Tables 4–6 are the same.

Table 4. Comparison of the performances of various models on the RSOD dataset.

Methods	*p*	R	AP				
			Aircraft	Oil Tank	Overpass	Playground	mAP (IOU = 0.5)
Faster R-CNN	0.873	0.748	0.859	0.867	0.882	0.904	0.878
SSD	0.824	0.682	0.692	0.712	0.702	0.813	0.729
RetinaNet	0.893	0.846	0.867	0.882	0.817	0.902	0.867
YOLOv3	0.850	0.693	0.743	0.739	0.751	0.852	0.771
YOLOv4	0.903	0.735	0.855	0.858	0.862	0.914	0.872
YOLOv5	0.897	0.872	0.873	0.884	0.854	0.932	0.886
DET-YOLO	**0.925**	**0.909**	**0.925**	**0.963**	**0.918**	**1.000**	**0.952**

Table 5. Comparison of the performances of various models on the UCAS-AOD dataset.

Methods	*p*	R	AP		
			Airplane	Car	mAP (IOU = 0.5)
Faster R-CNN	0.896	0.775	0.873	0.865	0.869
SSD	0.770	0.574	0.702	0.726	0.714
RetinaNet	0.887	0.742	0.843	0.865	0.854
YOLOv3	0.772	0.692	0.757	0.756	0.757
YOLOv4	0.894	0.732	0.857	0.862	0.859
YOLOv5	0.892	0.785	0.892	0.858	0.875
DET-YOLO	**0.962**	**0.863**	**0.997**	**0.892**	**0.945**

We plotted P-R curves and confusion matrices of DET-YOLO on the DOTA dataset to demonstrate the DET-YOLO detection performance for each category of the aerial targets. Figures 8 and 9 show the P-R curve and the confusion matrix, respectively.

Table 6. Comparison of the performances of various models on the UAVDT dataset.

Methods	p	R	AP			
			Car	Truck	Bus	mAP (IOU = 0.5)
YOLOv4	0.438	0.431	0.765	0.104	0.332	0.400
YOLOv5	**0.471**	0.427	0.767	0.121	0.349	0.419
DET-YOLO	0.464	**0.432**	**0.777**	**0.131**	**0.365**	**0.424**

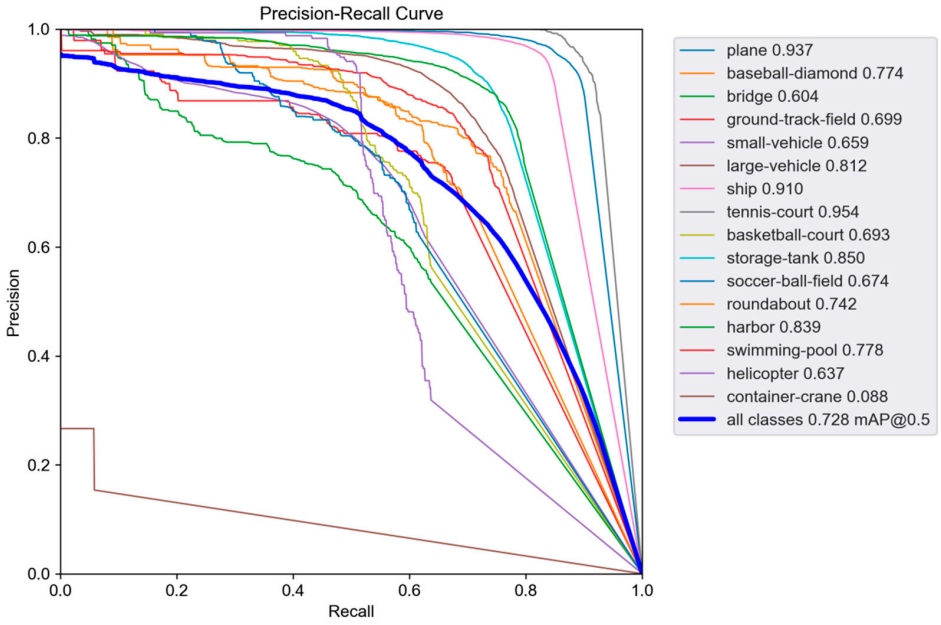
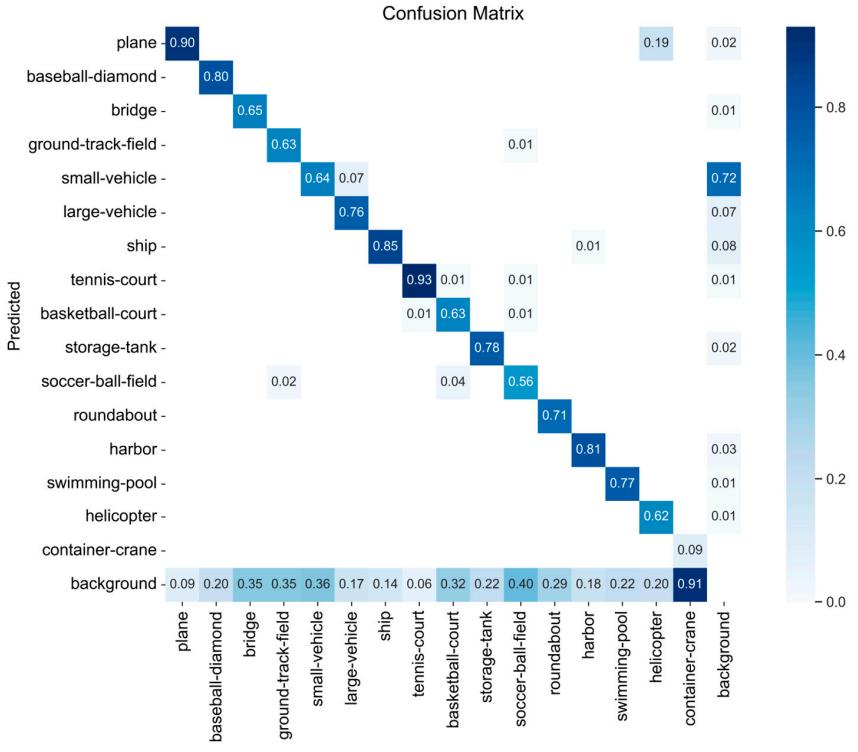
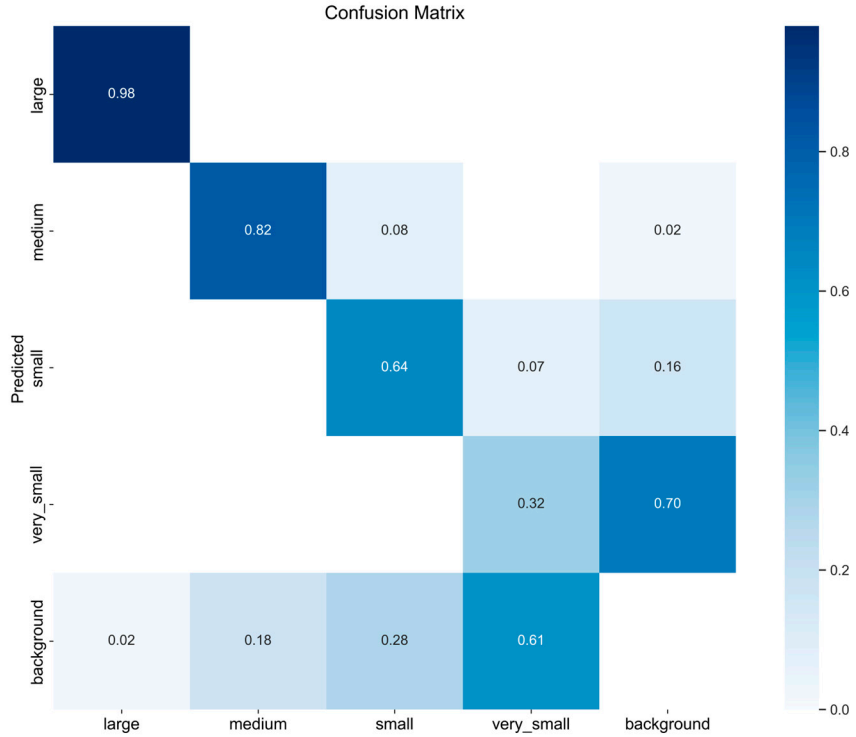


Figure 8. The P-R curve of DET-YOLO on the DOTA dataset at an IoU threshold of 0.45 and a confidence threshold of 0.25.



(**a**)

Figure 9. *Cont.*



(**b**)

Figure 9. Confusion matrix of DET-YOLO on the DOTA dataset at an IoU threshold of 0.45 and a confidence threshold of 0.25: (**a**) Confusion matrix based on the class of the target; (**b**) Confusion matrix based on the size of the target.

The AP of each category is the area below the corresponding P-R curve. The greater the AP, the better the detection performance for this category, and the larger the corresponding area. Each row of the confusion matrix represents an actual category of the sample, while each column represents the predicted category. The entire matrix reflects the model's ability to classify the objectives of each category. The horizontal background represents FP, the negative sample proportion of the prediction error, and the vertical background represents FN, the positive sample proportion of the prediction error. In the confusion matrix, the diagonal data indicate the proportions of correctly classifiable categories. From the P-R curve, it can be seen that the prediction accuracy for container cranes was far lower than that of the other categories. Moreover, the confusion matrix in Figure 9a reveals that the FN value corresponding to container cranes was the highest, indicating a severe leakage phenomenon. This was because the number of training samples with container cranes was significantly lower than the other training samples, and there were only 774 in 970,170 instances, which made it difficult for the model to fit the corresponding features and led to the model's poor feature extraction ability, making it difficult to effectively identify such targets. On the other hand, although the corresponding accuracy for small vehicles was great, the corresponding FP value was the highest, and there was a serious misdetection phenomenon. Small vehicles are small or extremely small targets, and there is a phenomenon known as stacking. Moreover, the cutting of the images caused some small vehicles in the dataset not to be marked, which further affected the FP value of the small vehicles. The confusion matrix in Figure 9b demonstrates that as the target became smaller, the prediction accuracy of the model decreased, with increasing leakage and misdetection.

We tested the reasoning speeds of different methods on the DOTA dataset, as shown in Table 7. Our method's speed was between YOLOv4 and YOLOv5, as can be seen. DET-YOLO's inference speed was slower than that of YOLOv4 because self-attentive operations are more computationally complex than convolutional operations. In addition, deformable embedding is a large kernel convolutional operation, which further slowed down DET-

YOLO's inference speed. Overall, our proposed DEViT required more computational resources and had greater feature extraction capability than YOLOv4's CSPDarknet53. The more complex feature extraction network is the primary reason why DET-YOLO's inference speed was slower than that of YOLOv4. YOLOv5 introduced a greater number of CSP structures in the neck than YOLOv4, resulting in a larger neck with more feature fusion capability and a reduction in the inference speed. DET-YOLO was faster than YOLOv5 because a large number of skip connection structures were discarded, and the use of FFCN sped up the calculation of DEViT. Future research should focus on how to enhance self-attentive modules and on embedding methods to increase speed.

Table 7. A comparison of the inference times for various methods on DOTA datasets.

Methods	Speed (ms per Picture)
YOLOv3	28.4 ms
YOLOv4	43.2 ms
YOLOv5	102.0 ms
TPH-YOLOv5	123.5 ms
DET-YOLO	62.1 ms

As shown in Figure 10, we visualized the prediction results on DOTA, RSOD, and UCAS-AOD and compared them with YOLOv4 and YOLOv5 to demonstrate the predictive effect of DET-YOLO.

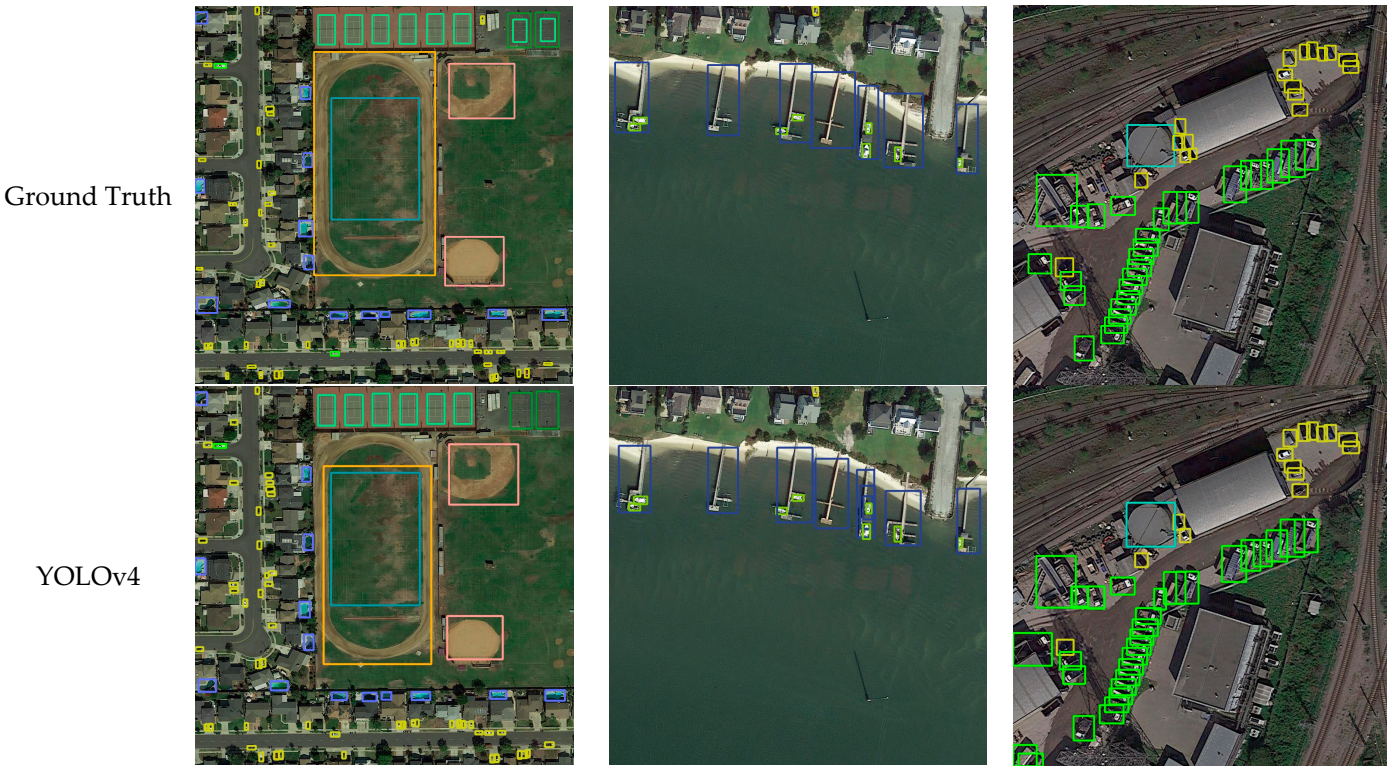
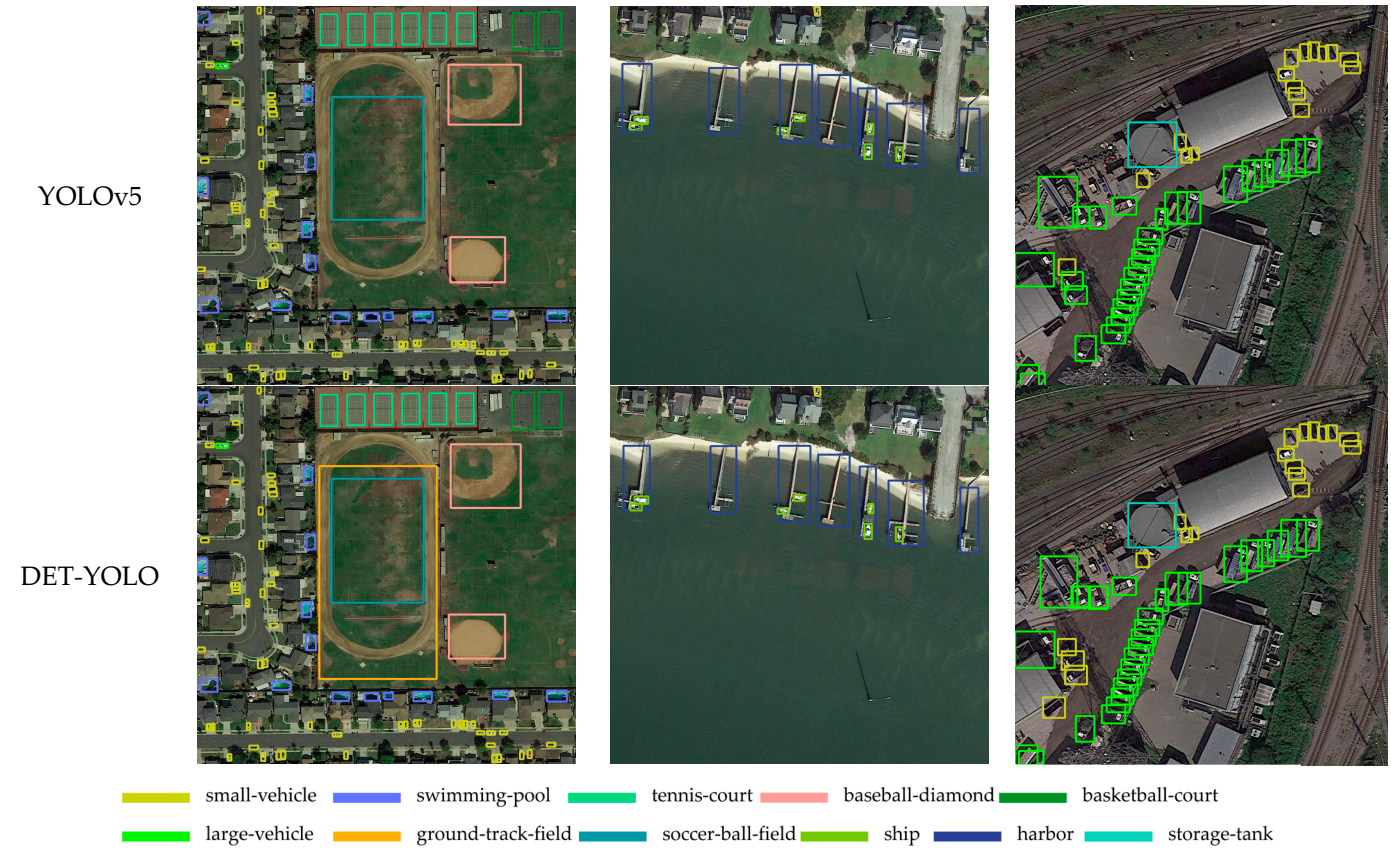


Figure 10. *Cont.*



(**a**)

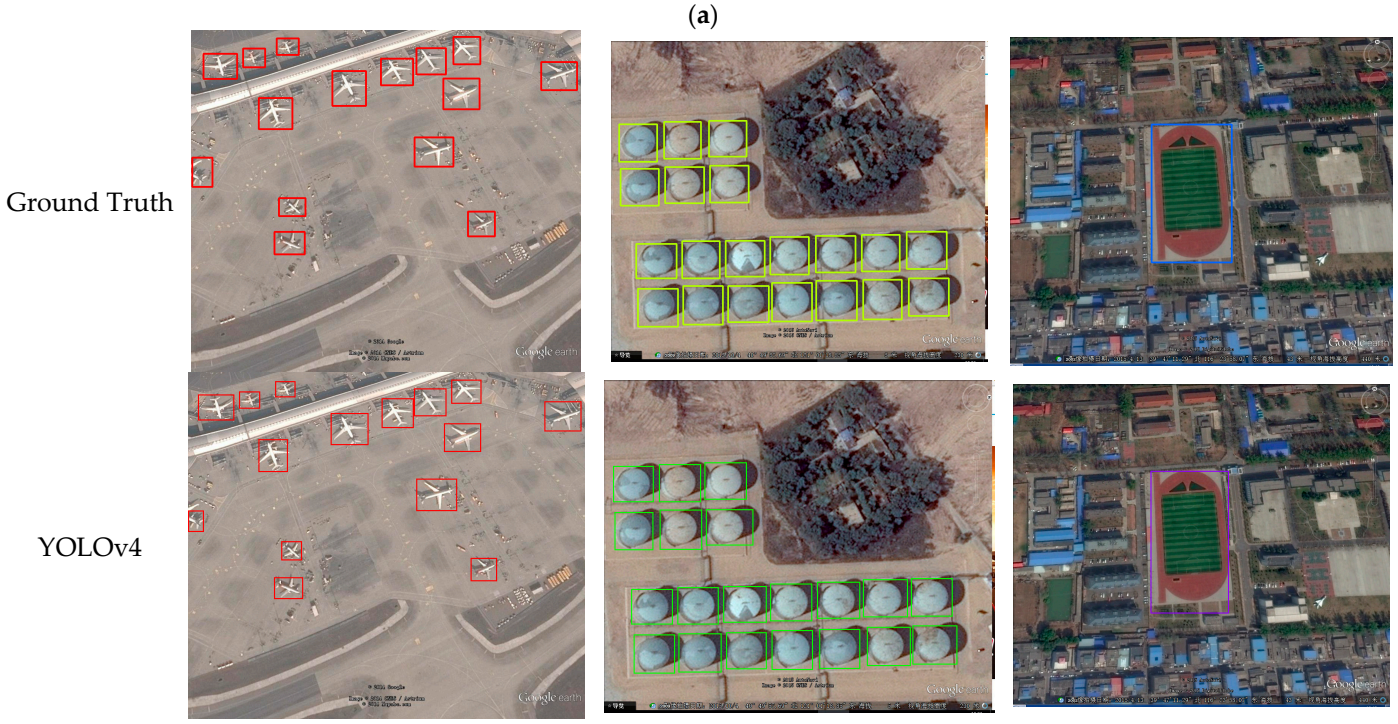


Figure 10. *Cont.*

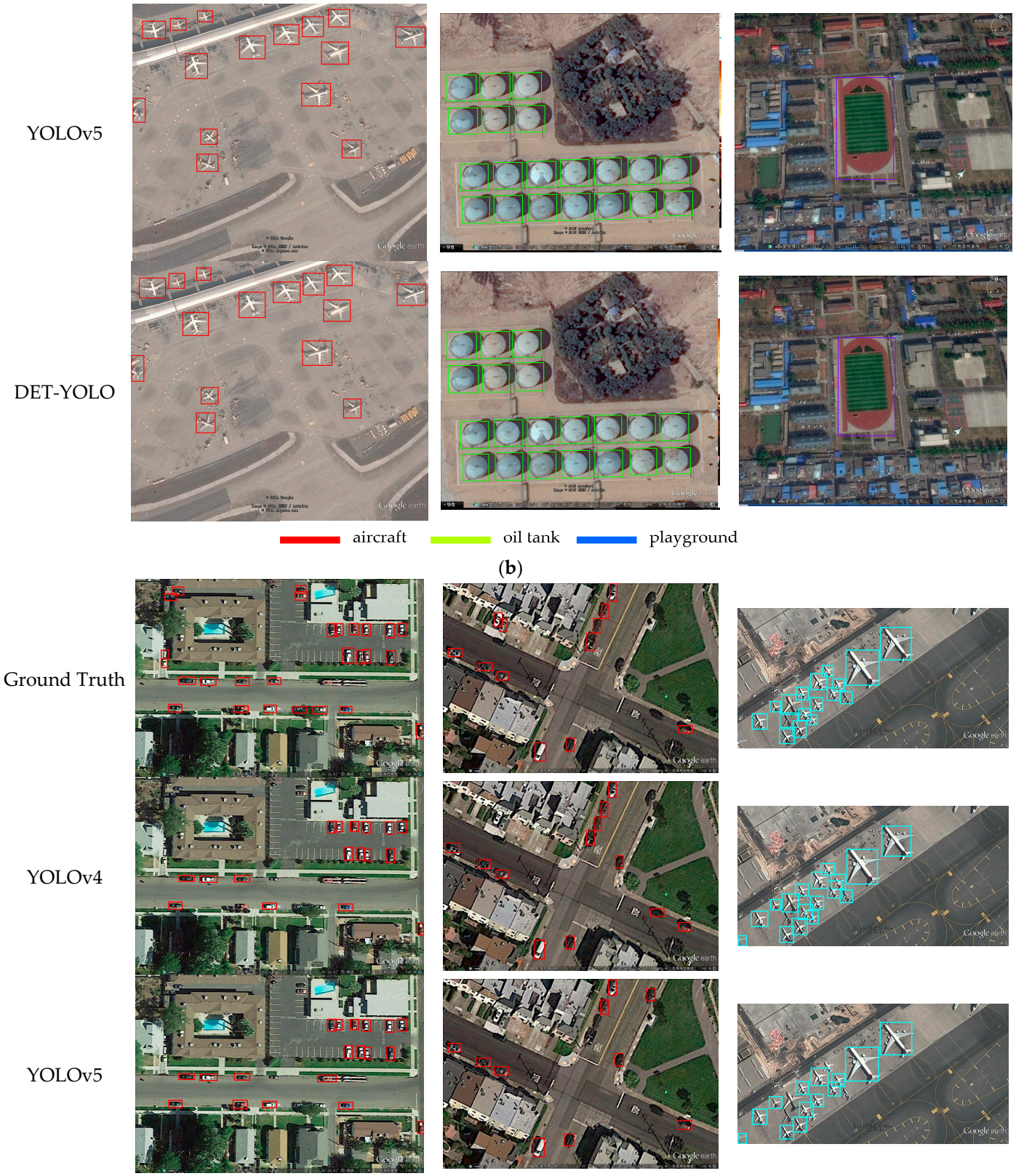
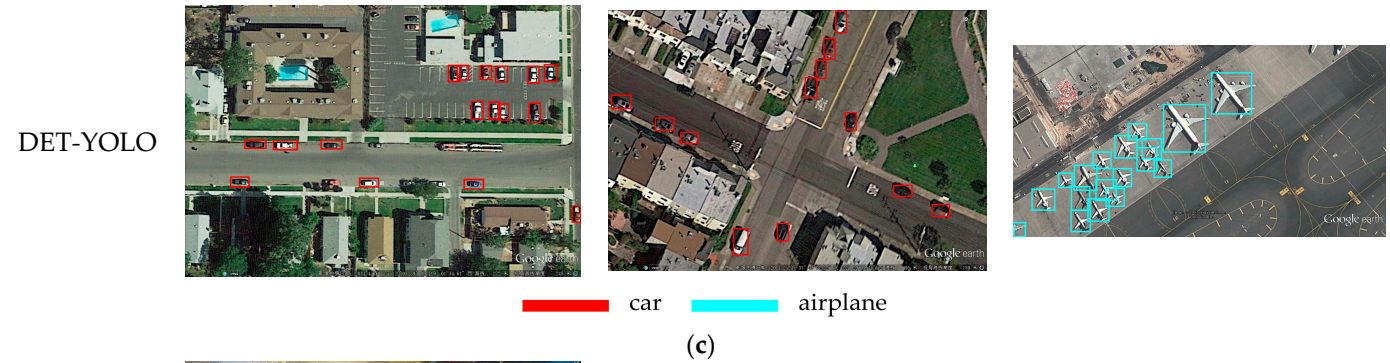
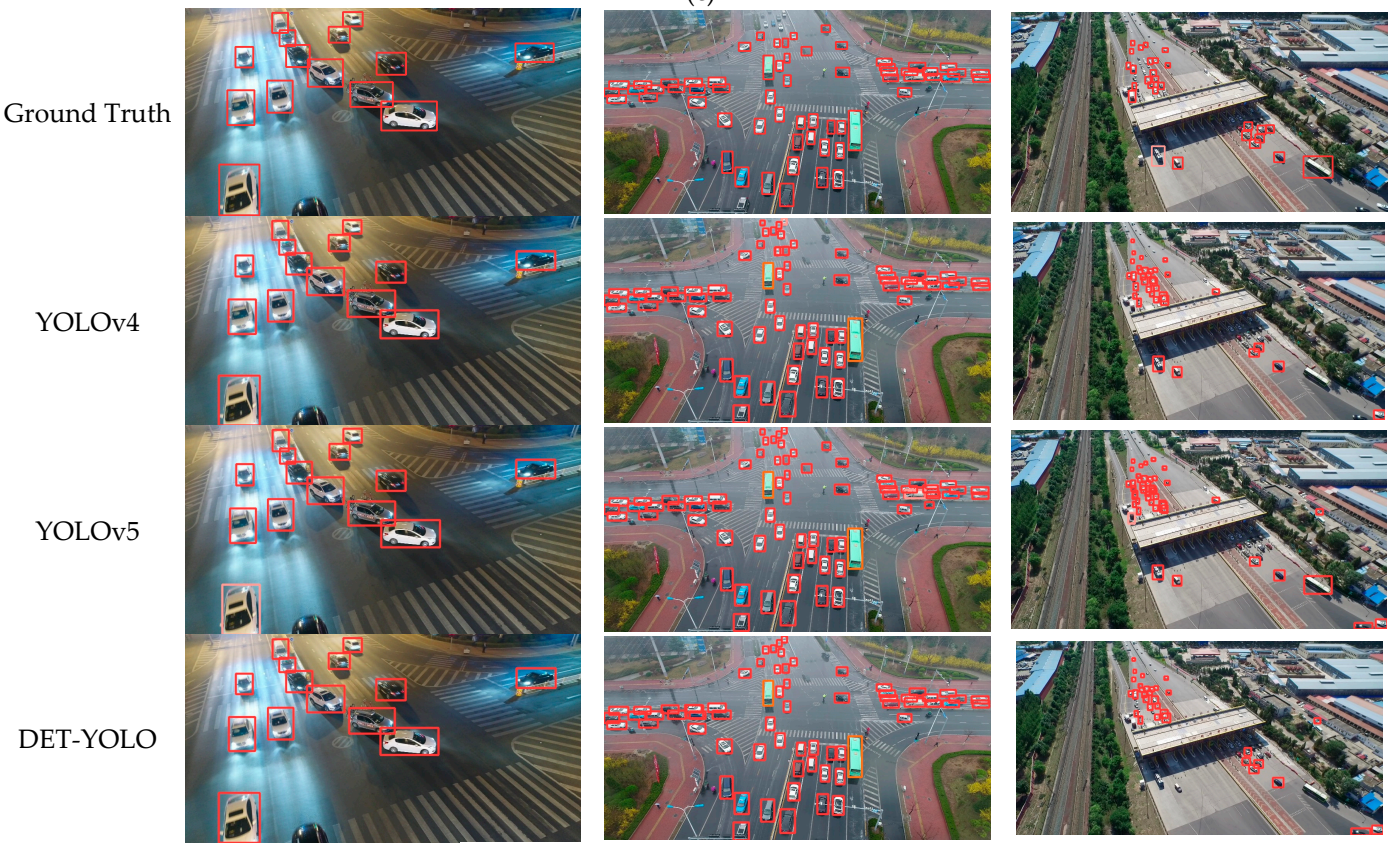


Figure 10. *Cont.*



(**c**)



(**d**)

Figure 10. Experimental results for (**a**) DOTA dataset; (**b**) RSOD dataset; (**c**) UCAS-AOD dataset; and (**d**) UAVDT dataset.

4.2. Ablation Experiments

In order to confirm the efficacy of the modifications we made, we conducted ablation experiments on each component; the results are presented in the Table 8. We used YOLOv4 as our baseline. After converting the backbone from cspdarknet53 to ViT, it was discovered that mAP decreased. This was due to the fact that ViT uses a crude linear embedding layer to convert images to high-dimensional embeddings, thereby destroying a substantial amount of effective feature information. After FPN was replaced with DSDP, the mAP increased by 0.014. This was a result of DSDP, which focuses the network on the target area and mitigates, to some extent, the target loss caused by cutting during the embedding process. However, after replacing linear embedding with deformable embedding, the mAP was greatly increased by 0.037. This was due to the fact that the cutting window could be deformed adaptively during the embedding procedure, effectively preserving the feature data. In comparison to ViT, which employs deformable embedding rather

than linear embedding, DEViT's spatial feature extraction capability was enhanced by its use of FCFN rather than FFN. Consequently, the mAP increased by 0.023. Using DSDP instead of FPN based on DEViT increased the mAP by 0.018, the same as before. We thoroughly demonstrated the effectiveness of the proposed modules. Among them, deformable embedding improved the mAP by approximately 0.04; followed by FCFN, which improved the model accuracy by approximately 0.02; and DSDP, which improved the model accuracy by approximately 0.02. In addition, we showed the effect of replacing FCFN with a self-attention transformation (SAT) and DSDP with a pyramid attention layer (PAL) in DET-YOLO, as shown in rows 6 and 7. The mAP using SAT was 0.717, which was a decrease of 0.011 compared to using FCFN. The mAP using PAL was 0.720, which was a decrease of 0.008 compared to DSDP.

Table 8. The results of ablation experiments on DOTA datasets.

Methods	ViT	DE	DEViT	FPN	DSDP	SAT	PAL	$mAP_{0.50}$	$mAP_{0.50:0.95}$
YOLOv4	√							0.653	0.438
DET-YOLO	√			√				0.650	0.433
	√				√			0.664	0.447
	√	√		√				0.687	0.471
			√	√				0.710	0.495
	√	√			√	√		0.717	0.509
			√				√	0.720	0.514
			√		√			0.728	0.515

5. Conclusions

On the basis of YOLOv4, we developed DET-YOLO to address the issues of diverse target sizes, complex background, and the large number of small targets in aerial images. We improved YOLOv4 from two perspectives, the network for feature extraction and the network for feature fusion, in order to better adapt it to the characteristics of aerial images. On one hand, we proposed DEViT to replace CSPDarknet53 as the network for feature extraction in order to improve the global capability for feature extraction. DEViT uses deformable embeddings instead of linear embeddings to effectively reduce the loss of feature information when processing targets at different scales compared to conventional vision transformers. We used FCFN rather than FFN in the DEViT encoder block to improve the extraction of location information while simultaneously reducing computational effort. On the other hand, we made a replacement for the FPN in the neck. As an alternative to the FPN, we proposed DSDC for extracting multiscale features from a single-scale feature map. DSDC is capable of adaptively focusing on key regions in order to enhance the ability to extract information about desired feature targets. DET-YOLO was evaluated using the widely used DOTAv1.5, RSOD, and UCAS-AOD datasets. For these datasets, the mAP values of the method proposed in this paper were 0.728, 0.952, and 0.945, which were superior to YOLOv4. The preceding results conclusively demonstrated the efficacy of the proposed DET-YOLO algorithm for object detection in aerial images. Experiments with ablation demonstrated that each of the proposed modules had a practical enhancement effect. In general, our proposed DET-YOLO model is a detector suited for aerial target detection tasks. In a subsequent work, we will enhance the linear embedding process of the transformer and attempt to apply deformable embedding to the hierarchical transformer model, thereby making the transformer model more suited to specific tasks in aerial imagery.

Author Contributions: Conceptualization, Y.W.; methodology, Y.W.; software and experiments, Y.W.; validation, Y.W.; writing—original draft preparation, Y.W.; writing—review and editing, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation (grant number 31574727) and the General Program of the Natural Science Foundation of Hunan Province (grant number 202049382).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Lee, J.; Moon, S.; Nam, D.W.; Lee, J.; Oh, A.R.; Yoo, W. A Study on the Identification of Warship Type/Class by Measuring Similarity with Virtual Warship. In Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Republic of Korea, 21–23 October 2020; pp. 540–542.
2. Reilly, V.; Idrees, H.; Shah, M. Detection and tracking of large number of targets in wide area surveillance. In *Computer Vision—ECCV 2010, Proceedings of the 11th European Conference on Computer Vision, Heraklion, Crete, Greece, 5–11 September 2010*; Daniilidis, K., Maragos, P., Paragios, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 186–199.
3. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 778–782. [CrossRef]
4. Yang, N.; Li, J.; Mo, W.; Luo, W.; Wu, D.; Gao, W.; Sun, C. Water depth retrieval models of East Dongting Lake, China, using GF-1 multi-spectral remote sensing images. *Glob. Ecol. Conserv.* **2020**, *22*, e01004.
5. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
6. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
7. Ren, S.Q.; He, K.M.; 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] [PubMed]
8. He, K.M.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
9. 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.
10. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
11. Redmon, J.; Farhadi, A.J. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
12. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Computer Vision—Eccv 2016, Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer: Cham, Switzerland, 2016; Volume 9905, pp. 21–37.
13. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv* **2017**, arXiv:1701.06659.
14. Ma, W.; Wang, X.; Yu, J. A Lightweight Feature Fusion Single Shot Multibox Detector for Garbage Detection. *IEEE Access* **2020**, *8*, 188577–188586. [CrossRef]
15. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Gelly, S. An image is worth 16 × 16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations, Virtual Event, 3–7 May 2021.
16. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the European Conference on Computer Vision (ECCV), Online, 23–28 August 2020; pp. 213–229.
17. Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; Joulin, A. Emerging properties in self-supervised vision transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 9650–9660.
18. Everingham, M.; van Gool, L.; Williams, C.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
19. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollar, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
20. Cheng, G.; Lang, C.; Wu, M.; Xie, X.; Yao, X.; Han, J. Feature enhancement network for object detection in optical remote sensing images. *J. Remote Sens.* **2021**, *2021*, 9805389. [CrossRef]

21. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
22. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. YOLOX: Exceeding YOLO Series in 2021. *arXiv* **2021**, arXiv:2107.08430.
23. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. Cspnet: A new backbone that can enhance learning capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
24. He, K.M.; Zhang, X.Y.; Ren, S.Q.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef] [PubMed]
25. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8759–8768.
26. Misra, D. Mish: A self regularized non-monotonic activation function. *arXiv* **2019**, arXiv:1908.08681.
27. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier Nonlinearities Improve Neural Network Acoustic Models. 2013. Available online: https://www.mendeley.com/catalogue/a4a3dd28-b56b-3e0c-ac53-2817625a2215/ (accessed on 1 June 2021).
28. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-iou loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
29. Zhang, Y.-F.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and Efficient IOU Loss for Accurate Bounding Box Regression. *arXiv* **2021**, arXiv:2101.08158. [CrossRef]
30. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
31. Wang, W.; Xie, E.; Li, X.; Fan, D.P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv* **2021**, arXiv:2102.12122.
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. *arXiv* **2021**, arXiv:2103.14030.
33. Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; Jégou, H. Training data-efficient image transformers & distillation through attention. In Proceedings of the International Conference on Machine Learning, Pasadena, CA, USA, 13–15 December 2021; pp. 10347–10357.
34. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 764–773.
35. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
36. Howard, A.; Zhu, M.; Chen, B.; Kalenichenko, D. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
37. Islam, M.; Jia, S.; Bruce, N. How much position information do convolutional neural networks encode. *arXiv* **2020**, arXiv:2001.08248.
38. Li, Y.; Mao, H.; Girshick, R.; He, K. Exploring plain vision transformer backbones for object detection. *arXiv* **2022**, arXiv:2203.16527.
39. Long, Y.; Gong, Y.; Xiao, Z.; Liu, Q. Accurate Object Localization in Remote Sensing Images Based on Convolutional Neural Networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2486–2498. [CrossRef]
40. Zhu, H.; Chen, X.; Dai, W.; Fu, K.; Ye, Q.; Jiao, J. Orientation robust object detection in aerial images using deep convolutional neural network. In Proceedings of the 2015 IEEE International Conference on Image Processing, Quebec City, QC, Canada, 27–30 September 2015; pp. 3735–3739.
41. 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.
42. Xia, G.S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. Dota: A large-scale dataset for object detection in aerial images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3974–3983.
43. Xu, D.; Wu, Y. Improved YOLO-V3 with DenseNet for Multi-Scale Remote Sensing Target Detection. *Sensors* **2020**, *20*, 4276. [CrossRef] [PubMed]
44. Xu, D.; Wu, Y. MRFF-YOLO: A Multi-Receptive Fields Fusion Network for Remote Sensing Target Detection. *Remote Sens.* **2020**, *12*, 3118. [CrossRef]
45. Jocher, G.; Nishimura, K.; Mineeva, T. Yolov5. Available online: https://github.com/ultralytics/yolov5 (accessed on 23 September 2022).
46. Prechelt, L. Early stopping—But when? In *Neural Networks: Tricks of the Trade*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 53–67.
47. Gong, H.; Mu, T.; Li, Q.; Dai, H.; Li, C.; He, Z.; Wang, W.; Han, F.; Tuniyazi, A.; Li, H.; et al. Swin-Transformer-Enabled YOLOv5 with Attention Mechanism for Small Object Detection on Satellite Images. *Remote Sens.* **2022**, *14*, 2861. [CrossRef]

48. Long, X.; Deng, K.; Wang, G.; Zhang, Y.; Dang, Q.; Gao, Y.; Shen, H.; Ren, J.; Han, S.; Ding, E.; et al. PP-YOLO: An effective and efficient implementation of object detector. *arXiv* **2020**, arXiv:2007.12099.
49. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone- captured scenarios. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Nashville, TN, USA, 19–25 June 2021; pp. 2778–2788.