*remote sensing*

MDPI

*Article*

# DCEF$^2$-YOLO: Aerial Detection YOLO with Deformable Convolution–Efficient Feature Fusion for Small Target Detection

**Yeonha Shin [1], Heesub Shin [2], Jaewoo Ok [2], Minyoung Back [2], Jaehyuk Youn [2] and Sungho Kim [1,*]**

[1] Advanced Visual Intelligence Laboratory, Department of Electronic Engineering, Yeungnam University, 280 Daehak-ro, Gyeongsan 38541, Republic of Korea; haa3551@ynu.ac.kr
[2] LIG Nex1 Co., Ltd., Yongin 16911, Republic of Korea; heesub.shin2@lignex1.com (H.S.); jaewoo.ok2@lignex1.com (J.O.); minyoung.back@lignex1.com (M.B.); jaehyuk.youn@lignex1.com (J.Y.)
* Correspondence: sunghokim@yu.ac.kr; Tel.: +82-810-3530

**Abstract:** Deep learning technology for real-time small object detection in aerial images can be used in various industrial environments such as real-time traffic surveillance and military reconnaissance. However, detecting small objects with few pixels and low resolution remains a challenging problem that requires performance improvement. To improve the performance of small object detection, we propose DCEF$^2$-YOLO. Our proposed method enables efficient real-time small object detection by using a deformable convolution (DFConv) module and an efficient feature fusion structure to maximize the use of the internal feature information of objects. DFConv preserves small object information by preventing the mixing of object information with the background. The optimized feature fusion structure produces high-quality feature maps for efficient real-time small object detection while maximizing the use of limited information. Additionally, modifying the input data processing stage and reducing the detection layer to suit small object detection also contributes to performance improvement. When compared to the performance of the latest YOLO-based models (such as DCN-YOLO and YOLOv7), DCEF$^2$-YOLO outperforms them, with a mAP of +6.1% on the DOTA-v1.0 test set, +0.3% on the NWPU VHR-10 test set, and +1.5% on the VEDAI512 test set. Furthermore, it has a fast processing speed of 120.48 FPS with an RTX3090 for 512 × 512 images, making it suitable for real-time small object detection tasks.

**Keywords:** aerial object detection; small target detection; real-time object detection; DCN; deformable convolution; feature fusion



**Citation:** Shin, Y.; Shin, H.; Ok, J.; Back, M.; Youn, J.; Kim, S. DCEF$^2$-YOLO: Aerial Detection YOLO with Deformable Convolution–Efficient Feature Fusion for Small Target Detection. *Remote Sens.* **2024**, *16*, 1071. https://doi.org/10.3390/rs16061071

Academic Editor: Hossein M. Rizeei

Received: 23 January 2024
Revised: 2 March 2024
Accepted: 15 March 2024
Published: 18 March 2024



## 1. Introduction

Deep learning technology is used ubiquitously in industry and is developing at an incredible rate. Therefore, the performance of simple classification and detection tasks using representative benchmark datasets such as ImageNet [1], COCO [2], and Pascal VOC [3] is saturated. Accordingly, much research is being conducted in each field to perform special and applied tasks depending on the purpose of use. Problem area diagnosis and segmentation using medical images [4,5], small object detection using aerial imagery datasets [6,7], and object detection in SAR military imagery [8,9] are examples.

As the performance of SOTA models becomes saturated, they have high detection rates and accuracy in images that are not very complex. Nevertheless, performance improvement is still needed for partially overlapping objects or small objects. When 2D images and videos are used, small objects, having a limited number of pixels and low resolution, provide insufficient information for deep learning models to extract features through convolution operations. In particular, in the case of aerial image datasets such as DOTA [10], compared to the very high resolution of up to 20,000 × 20,000, the spatial proportion of small objects is very small, as low as 10 pixels, and simple resizing and pooling operations performed during the learning process can cause information loss in small objects. Transformer-based models [11,12] have recently performed well and are topping the benchmark leaderboard.

On the other hand, these models, which divide images into patches and perform attention operations [13–15], may not be suitable for detecting small objects.

Accordingly, this study examines detection rate improvement from two major perspectives to detect small objects of 32 × 32 pixels or smaller in long-distance aerial images. Therefore, we propose aerial detection YOLO with deformable convolution-efficient feature fusion for small target detection (DCEF$^2$-YOLO; Deformable Convolution-Efficient Feature Fusion-YOLO).

Make full use of the internal feature information of small objects in a few pixels: The deformable convolution (DFConv) module was used for this purpose [16,17]. The convolution kernel is square in shape, so the object and surrounding background information may be mixed during operation. This can be fatal when learning tiny objects with a small amount of information and can be compensated for using the DFConv module, which allows the shape of the kernel to be modified. DFConv acquires the offset values through additional kernel training to adjust the sampling points of the kernel and performs sampling feature masking according to learning contribution through sampling weight. This was proposed to ensure that the kernel has an appropriate receptive field by training it to be robust to various forms of geometric transformations caused by camera angles, object posture or state, location, etc. And we expect to improve learning for small objects by preventing mixing with background information and focusing on areas of interest more effectively than possible with the original kernels. And several previous studies dealing with small object detection have confirmed that this actually leads to performance improvements. Ref. [18] improved the detection accuracy of thin and rotated targets during remote sensing by replacing all convolution layers of YOLOv5 with DFConv and adding the box aspect ratio to the loss function to improve the detection accuracy for specific classes. In addition, in [19,20], DFConv was applied to YOLOv5 to detect small objects in complex environments, and channel-level attention was applied through a CBAM (convolutional block attention module), resulting in performance improvement. In this paper, in order to prevent an increase in the number of calculations, we found the optimal location and number of each module through experiments and applied them to the network to improve performance with minimal changes and applications.

Must be usable for real-time tasks: Small object detection in aerial images is expected to be used mainly in surveillance/reconnaissance and military fields, requiring onboard real-time task completion. Configuring the network to be complex in order to have good performance limits real-time operations. Therefore, YOLOv5 [21] was used as the base model because it is lightweight but has good detection performance and scalability. In addition, Efficient-RepGFPN [22], which performs various stages of feature fusion, was applied to make the most of the small object feature information extracted from the backbone. Unlike the YOLOv5 series, which perform relatively simple fusions in the neck, Efficient-RepGFPN performs sufficient inner fusion between multiple and same scales of feature maps to extract more meaningful information. However, this naturally brings a lot of computational cost and is not suitable for real-time tasks. Therefore, we optimize the structure considering performance and computational cost to suit our purpose and improve small object detection performance without computational overhead. In addition, the focus module of YOLOv5 was replaced with convolution-based processing, which is faster on a GPU, and the deformable convolution used in the backbone does not pose a large computational burden. Based on this, we propose a DCEF$^2$-YOLO that is suitable for real-time tasks.

The contributions are summarized as follows:

(1) To make the most of a small amount of small object internal information, DCN and an optimized efficient feature fusion neck are applied. This actually improves the small object detection performance in the benchmark aerial dataset without significant computational burden. DCEF$^2$-YOLO improves mAP by 6.1% in DOTA, mAP by 1.5% in VEDAI, and mAP by 0.3% in NWPU VHR-10 compared to the latest comparison model.

(2) Each module and structure is applied through an appropriate number and location selection and optimization process in consideration of the amount of calculation for real-time tasks. Accordingly, DCEF$^2$-YOLO has the smallest amount of calculations (GFLOPs) compared to the comparative models, and shows a detection speed of 120.48 FPS on 512 × 512 images.

Section 2 introduces the recent research trends related to small object detection in aerial images and briefly explains YOLOv5, which was selected as the base model. Section 3 explains the DCN module, optimization of Efficient-RepGFPN, and the focus to conv function. Section 4 evaluates the network performance using several public datasets, and Section 5 presents the conclusions.

## 2. Related Works

### *2.1. Aerial Object Detection*

Object detection work is divided mainly into a one-stage detection and a two-stage detection depending on the task execution stage, and it is also divided depending on the bounding box type: HBB (horizontal bounding box)-based or OBB (oriented bounding box)-based detection. OBB is a form in which the box direction parameter (a rotation component) is added to the HBB coordinates (x, y, w, h, and theta). This system is applied mainly as an extension of the HBB-based two-stage detector and can be more suitable for rotated objects. OBB-based detection models have been used widely in recent years because aerial object detection mainly utilizes datasets containing unaligned objects captured from the air.

Pu et al., proposed the ARC (adapted rotated convolution), which adaptively rotates the convolution kernel according to the object direction and increases the identification and detection rate of objects in various directions [23]. Yi et al., proposed a box-boundary-aware vector learning method that broke away from the existing method of directly predicting the width, height, and angle of the bounding box and showed a good rotated object detection rate [24]. Yang et al., presented SkewIOU loss based on Gaussian modeling and the Gaussian product for rotated object detection, and Wang et al., introduced FCOSR Assigner and ProbIOU Loss to detect rotated bounding boxes, and a decoupled angle prediction head predicted the angle distribution of the object more accurately [25,26].

Previous studies showed good detection performance while using existing HBB. Kim et al., performed attention at the channel level to compensate for the lack of small object information and used transposed convolution to prevent small object information from mixing through a simple upsampling process [6]. Zhang et al., exploited the characteristics of the VEDAI dataset, which provides RGB and IR images of two sizes (512, 512) and (1024, 1024), and reported excellent detection performance through multi-sensor image fusion and super-resolution loss [7,27]. Li et al., mentioned that different objects require different degrees of context and proposed LSKNet: a backbone model that dynamically adjusts the receptive fields through large-scale kernels and channel-specific pooling [28]. Wang et al., conducted optimization to pre-train an aerial dataset from the backbone, and the performance in downstream tasks was presented as an experiment [29].

Figure 1 summarizes the aerial object detection technologies listed above in a timeline. As recent trends show, many models based on OBB have been proposed. But OBB-based models have limitations in that they are difficult to fine-tune for new tasks due to the data label characteristics. And they are not good at distinguishing occluded objects with various shapes by considering angle information. Accordingly, we perform an HBB-based detection task considering high complexity, computational cost, and scalability. And through verification, it presents performance that is comparable to OBB-based detection tasks.

A publicly available aerial image dataset for detection is very helpful for optimizing and developing deep-learning models for small object detection tasks. Benchmark datasets include DOTA [10], HRSC2016 [30], and DIOR-R [31], as well as VEDAI [32], AID [33], xView [34], iSAID [35], LandCover.ai [36], NWPU VHR-10 [37], and VisDrone [38]. This

study examined the performance of DCEF$^2$-YOLO on DOTA, VEDAI, and NWPU VHR-10. Table 1 presents a summary of the utilized datasets.

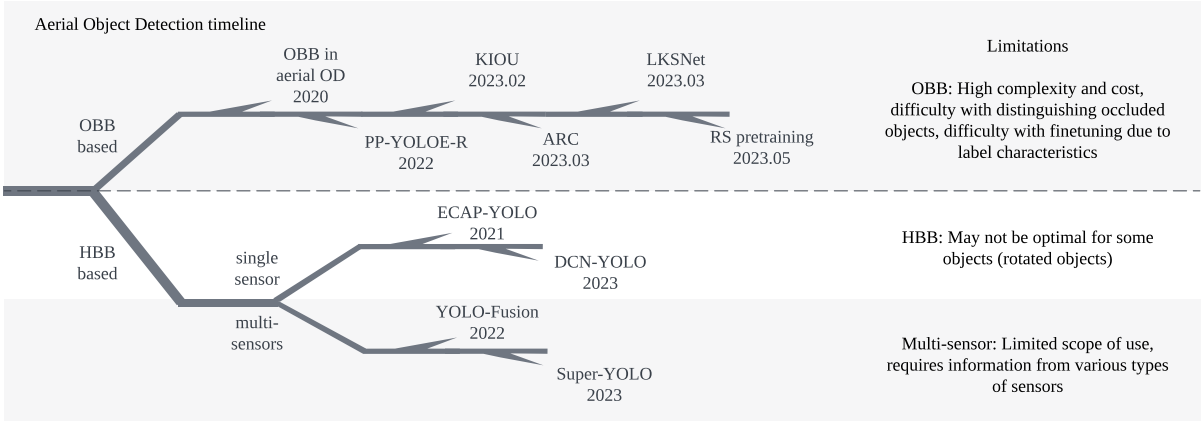



**Figure 1.** Aerial object detection timeline of the latest aerial object detection technologies according to their release dates. Depending on the bounding box format, methods are largely divided into OBB-based and HBB-based detection. The limitations of each technique are presented on the right. Several limitations of OBB were considered, and a model based on HBB was proposed for efficient real-time aerial image detection [6,7,23–29,39].

**Table 1.** DOTA-v1.0, VEDAI, and NWPU VHR-10 dataset summary. The DOTA, VEDAI, and NWPU VHR-10 datasets were used to verify the proposed model. The overall performance of DCEF$^2$-YOLO was verified through DOTA-v1.0 and NWPU VHR-10, which include multi-sized objects. The small object detection performance was analyzed more intensively through VEDAI, which includes small objects [40].

| | **DOTA-v1.0 [10]** | **VEDAI [32]** | **NWPU VHR-10 [37]** |
|---|---|---|---|
| Dataset size | 2806 images | 1272 images | 800 images |
| Image size | (800, 800) to (20,000, 20,000) | (512, 512), (1024, 1024) | Width and height range from 450 to 1000 |
| Target classes | vehicles, track fields, storage tanks, etc. | multi-class vehicles and others | vehicles, ships, planes, etc. |
| Target Size | multi sizes; l, m, s (less than 10 pixels) | small instances with widths of 8 to 20 pixels | multi sizes; l, m, s (primarily medium-sized) |

### *2.2. YOLOv5*

The YOLO series is a representative model of one-stage detectors [41]. Although the detection accuracy is slightly lower than that of two-stage detectors that perform object location estimation and classification separately [42,43], real-time application is possible with rapid detection speed. Since YOLOv5, there has been a focus on optimizing algorithms and training methodologies, leading to improvements and expansions, and in January 2023, YOLOv8 [43] was announced. DCEF$^2$-YOLO, which is optimized for small object detection through a combination of various technologies, was constructed based on the pure and highly scalable YOLOv5.

Like YOLOv4 [44], YOLOv5 distributes the computational load of each layer evenly through CSPNet [45], eliminates computational bottlenecks, and derives the computational utilization of the CNN layer. The detection performance is improved by approximately 10%, but it has a lower capacity and faster speed. The backbone receives an input image that is expanded and resized to the channel dimension through a focus operation, and performs learning on the image. PAFPN is applied to complement the unidirectional fusion flow of FPN; it fuses the feature maps received from the three stages of the backbone to

produce a more meaningful map for detection. YOLOv5 showed a detection performance of 68.9 mAP@0.5 on COCO 2017 val, and the smallest YOLOv5n model has a speed of up to 45 ms on a CPU. The model has various sizes up to n, s, m, l, and x according to the depth–width multiple. This study applied a YOLOv5s structure with suitable performance and high speed. Figure 2 shows the structure of YOLOv5s.

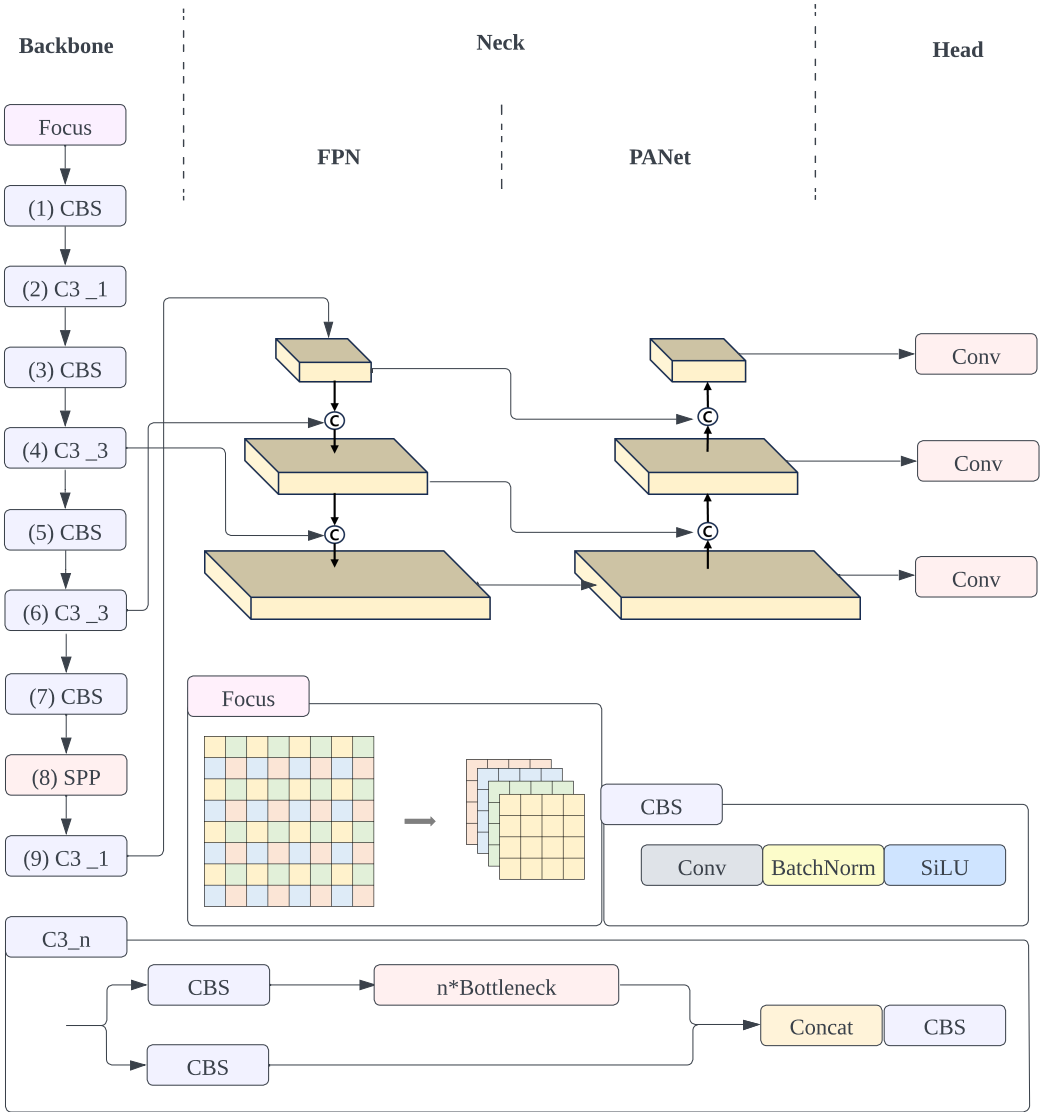



**Figure 2.** Overall structure of YOLOv5s. YOLOv5 evenly distributes the computational amount of each layer through CSPNet, eliminates the computational bottlenecks, and induces computational utilization of CNN layers. PAFPN generates more meaningful feature maps through bidirectional feature map fusion. Each layer of the backbone is numbered from (1) to (9), providing a reference to understand the model improvements for the proposed model.

## 3. Methods

Figure 3 presents the overall structure of DCEF$^2$-YOLO. Based on the YOLOv5s model, the focus module that processes the input image was replaced with convolution-based processing. A deformable convolution module was applied in the backbone to extract meaningful small object features. The relatively simple existing neck was replaced with an optimized version of Efficient-RepGFPN, which undergoes sufficient fusion. These improvements enable small object detection in aerial images with high real-time performance. Section 3.1 explains the DFConv module. Sections 3.2 and 3.3 explain Optimized-Efficient-RepGFPN, and focus to conv, respectively.

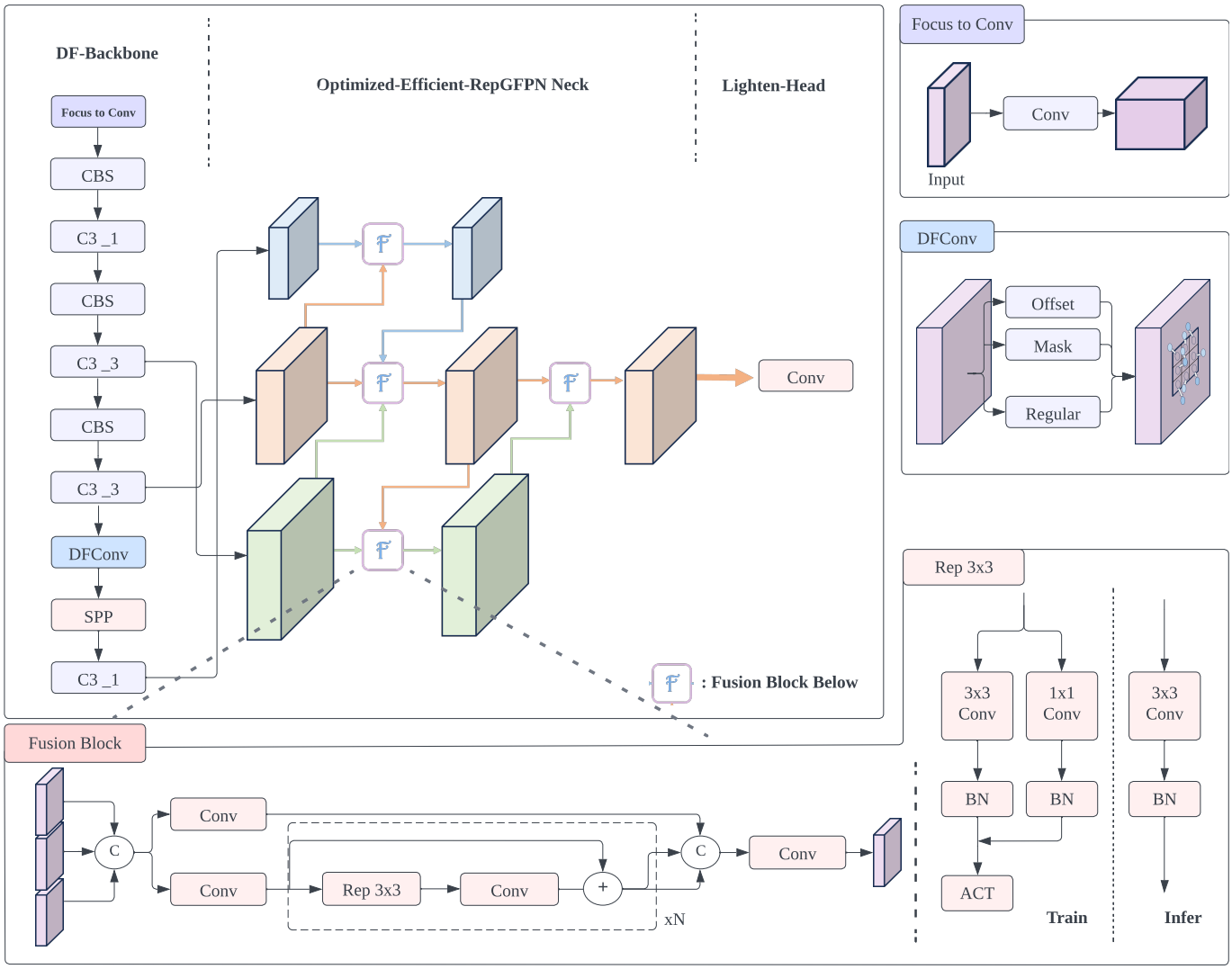



**Figure 3.** Overall structure of DCEF$^2$-YOLO. The upper left is the overall structure of the model. Each module is listed from the upper right to the bottom. DCEF$^2$-YOLO consists of a backbone, an Optimized-Efficient-RepGFPN neck, and a lighten head. When an image is an input, the input image is reconstructed through the replaced convolution-based input processing stage and passed to the backbone. In the backbone, the deformable convolution layer acquires meaningful inner-feature information for small objects by simultaneously learning offset, mask, and feature information through three kernels. Afterwards, Optimized-Efficient-RepGFPN, which was reduced to suit real-time work, performs sufficient fusions to make the most of the feature information extracted from the backbone. Efficient-RepGFPN performs feature map fusion through the fusion block, and the fusion block is shown at the bottom of the diagram. It has the structure of CSPNet and has rep 3 × 3 operations to reduce unnecessary calculations during inference. The rep 3 × 3 structure is an optimization technique that reduces inference time efficiently by separating the kernels used during training and inference. More details about Efficient-RepGFPN are described in Section 3.2. Additionally, the CBS layers and C3_n layers included in the backbone are the same as those in YOLOv5, as illustrated in Figure 2.

### *3.1. DCN*

Figure 4 presents the operation of deformable convolution. Unlike basic convolution, which uses a 3 × 3 square kernel (first from the left), DCNv1 (second from the left) uses the sampling offset to adjust the sampling position, and DCNv2 (second from the right) uses the sampling weight to improve the detection accuracy. The first picture on the right is an

example of a kernel with a sampling offset applied, which was the output after training in DCEF$^2$-YOLO.

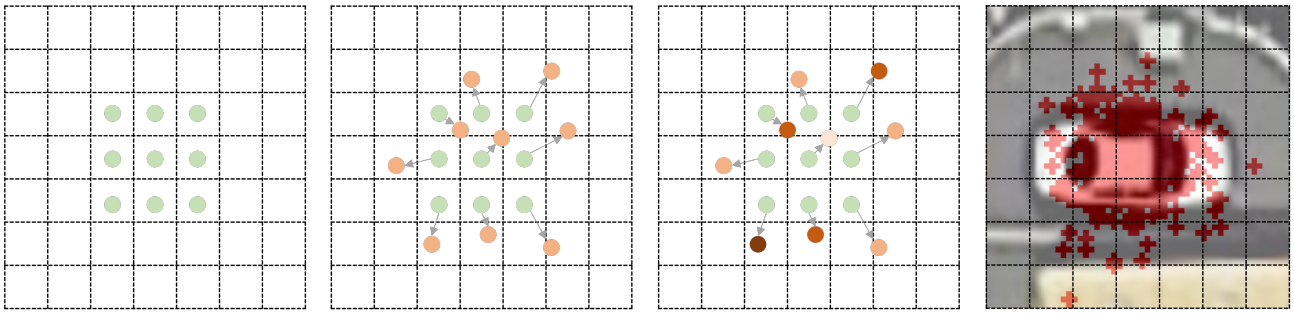


**Figure 4.** Example of deformable convolution kernel operation. Unlike the existing convolution, which utilizes a 3 × 3 square filter (first from the left), DCNv1.0 (second from the left) introduces a sampling offset value to modify the sampling position, and DCNv2.0 (second from the right) introduces a sampling weight that assigns a weight to each sampling point. The first picture on the right is an example of a filter with a sampling offset applied, which was the output after training in DCEF$^2$-YOLO.

Deformable convolution (DFConv) was used to derive meaningful small object feature extraction in the YOLOv5 backbone. DCNv1 [16] presented the first deformable convolution–deformable ROI (region of interest) pooling concept that utilizes a learnable sampling offset. Deformable convolution obtains an offset value through additional kernel training performed simultaneously with the existing kernel training and performs a transformation of the sampling position of the kernel. Through this, it fits the object more closely than basic convolution and becomes robust against geometric transformations. On the other hand, even if a deformable offset is given, the sampling space may still deviate from the area of interest. By DCNv2 [17], an additional sampling weight value is introduced to compensate for this. The sampling weight is also obtained through additional kernel training, where a large value is weighted if the sampled location significantly influences learning the correct answer by belonging to the characteristics of the object, and conversely, a small value is weighted. Deformable convolution can be confirmed by the following formula.

Assuming a 3 × 3 kernel with a dilation of 1, the output feature map for each position through a general convolution operation has the following values:

$$y(p) = \sum_{k=1}^{3\times3} \omega_k \cdot x(p + p_k) \tag{1}$$

where $p_k = (-1, -1), (-1, 0), \ldots, (1, 1)$, which is the existing kernel sampling location; $x$ is the input feature map; $y$ is the output feature map; $\omega_k$ is the weight for each sampling position. The output of the DCNv2 kernel reflecting the sampling offset and sampling weight is as follows:

$$y(p) = \sum_{k=1}^{3\times3} \omega_k \cdot x(p + p_k + \Delta p_k) \cdot \Delta m_k \tag{2}$$

$$x(p') = \textstyle\sum_q G(q, p') \cdot x(q), \ \textit{where}\ p' = (p + p_k + \Delta p_k), \\ q\ \textit{enumerates all integral spatial locations in the feature map}\ x \tag{3}$$

$$\sum_q G(q, p') = g(q_x, p'_x) \cdot g(q_y, p'_y)\ , \textit{where}\ g(a, b) = max(0, 1 - |a - b|) \tag{4}$$

The DCNv2 kernel multiplies the output from $p + p_k + \Delta p_k$, which reflects the sampling offset $\Delta p_k$, by the sampling weight $\Delta m_k$, which is between 0 and 1. At this time, $\Delta p_k$ is a fractional value obtained through a bilinear interpolation kernel $G(.,.)$. The sampling

offset and sampling weight are obtained through separate convolution training, and the values of the 3K channels are finally output: 2K channels with learnable offsets in the x- and y-directions and K channels with sampling weight scalar values that can be obtained through a sigmoid function.

We use a DCNv2 kernel (DFConv in this paper) to derive meaningful small object feature extraction from the YOLOv5 backbone. As shown in Table 2, when the DFConv module was applied, the recall improved by 3.2%, and the classification precision and mAP@0.5 improved by 3.3% and 2.2%, respectively. However, during the experiment, we discovered that stable learning can be difficult in small object detection tasks if an appropriate convergence direction is not found in the early stages of learning. This is presumed to be derived from the fact that parallel training is performed using two kernels (offset and existing feature extraction) simultaneously and that object information is lacking. We confirmed that applying DFConv at the layer immediately before SPP after learning enough about the input, rather than applying many DFConv modules, is the most stable method and results in sufficient performance improvement. Table 3 presents a performance comparison based on the location and number of DFConv modules. The backbone of YOLOv5 consists of nine layers, excluding the input stage, as shown in Figure 2. The labels (3), (5), and (7) in the table indicate the application positions of the DFConv module. We applied one to three DFConv modules and compared their performance. In Table 3, when one DFConv module was located in the (7) layer immediately before the SPP, the F1 score, which is an indicator of the precision–recall balance, was the highest, and the overall performance was the best, so stable training and general-purpose application to new data were possible. On the other hand, the model showed a tendency to diverge when three layers were replaced.

**Table 2.** Ablation table: w/, w/o DFConv. Performance comparison table for applying the DFConv module. The deformable convolution is effective for learning small object features.

| | **Precision** | **Recall** | **mAP@0.5** | **mAP@0.5:0.95** |
|---|---|---|---|---|
| w/o DFConv | 0.629 | 0.590 | 0.626 | 0.364 |
| w/DFConv | **0.662** | **0.622** | **0.648** | **0.372** |

**Table 3.** Ablation Table 2: position of DFConvs. Performance comparison table of DFConv modules by location. The most balanced performance improvement was achieved when one DFConv module was located in the (7) layer immediately before the SPP.

| | **Precision** | **Recall** | **mAP@0.5** | **mAP@0.5:0.95** | **F1 Score** * |
|---|---|---|---|---|---|
| (3) DFConv | 0.580 | **0.558** | 0.531 | 0.292 | 0.569 |
| (5) DFConv | 0.550 | 0.489 | 0.492 | 0.248 | 0.518 |
| (7) DFConv | **0.626** | 0.530 | **0.546** | **0.30** | **0.574** |
| (3) (5) DFConv | **0.648** | 0.464 | 0.526 | 0.293 | 0.526 |
| (3) (7) DFConv | 0.617 | **0.505** | **0.552** | **0.302** | **0.555** |
| (5) (7) DFConv | 0.538 | 0.485 | 0.492 | 0.277 | 0.510 |

* F1-score = $\frac{2}{(1/Precision)+(1/Recall)}$.

### *3.2. Efficient-RepGFPN*

We adopt the Efficient-RepGFPN structure, which performs sufficient feature map fusion to make full use of the small object information extracted through the DFConv module. Efficient-RepGFPN improves detection performance through sufficient inner fusions between multi/same-scale feature maps, but this naturally leads to a large increase in the amount of computation. Accordingly, we optimize the structure by reducing the neck and detection head to suit real-time small object detection tasks. DCEF$^2$-YOLO performs detection using only one feature map that has undergone sufficient fusion, which efficiently

detects small objects without imposing a computational burden compared to detection with the three feature maps of the existing YOLOv5.

GFPN (Generalized-FPN) [46], the base of Efficient-RepGFPN, is a structure proposed to compensate for the fact that FPN [47], PAFP [48], and BiFPN [49] only focus on the fusion of feature maps of different resolutions and lack connections between internal layers. A deep network was formed without significantly increasing computational complexity by adjusting the shortest gradient distance through log2n-link. Moreover, it fuses all feature maps from the current, previous, and subsequent layers through queen fusion by going beyond the existing structure that only performs feature map fusion between the current and previous layers. RepGFPN provides feature maps containing rich information to the detection head through sufficient feature fusion but was limited in its use in real-time applications. Therefore, DAMO-YOLO proposes Efficient-RepGFPN (called E-RGFPN), which optimizes GFPN and improves its performance.

E-RGFPN improves GFPN in three aspects. (1) Adjustment of channel depth by scale: GFPN unifies the channel depth of feature maps for each scale. In E-RGFPN, however, higher performance was achieved at a flexible channel depth of (96, 192, 384) by considering the trade-off between the channel depth for each scale and the fusion bottleneck width. (2) Adjustment of the number of upsampling operators: The number of upsampling operators in the overall structure was adjusted, suggesting that the upsampling operation and the corresponding connection layer do not significantly improve performance compared to the delay time. (3) Fusion block improvement: Convolution-based fusion was replaced with CSPNet [45], and re-parameterization and ELAN were applied like in YOLOv7 [50] to improve the accuracy without increasing the inference cost. Re-parameterization [51,52] optimizes the shared and task-specific weights through two convolutions. For task-specific loss, only the relevant weights were optimized to reduce the interference between tasks, and at the inference time, only shared weights optimized for joint indicators of interest were used, allowing for general-purpose application to a wide range of data without increasing the inference costs. ELAN [53] designed an efficient propagation path hierarchical aggregation structure by controlling the longest and shortest gradient paths, with the view that a shorter gradient length indicates more powerful network learning. This solved the problem of low convergence when expanding the network and amplified the usability of gradients.

DCEF$^2$-YOLO uses only the medium-sized output feature map through E-RGFPN for detection. Since the final output feature map that passes the backbone has a considerably lower spatial resolution than the input, by combining it with a low-level feature map, the information can be meaningfully utilized in a wider image unit at detection. At this time, the size of the object that is mainly referenced and detected varies according to the size of the feature map. The feature maps with small sizes and deep channels through several layers mainly contribute to detecting large objects and learning the overall context of the image. And the feature maps with relatively large sizes and shallow channels containing the wide part of the image contribute to small object detection. Therefore, in some networks for which primary purpose is small object detection, medium or larger object detection layers are removed for efficient and intensive training [6,7].

In Figure 5, Efficient-RepGFPN finally outputs feature maps with three scales: s, m, and l. In this case, the l-scale feature map has relatively less fusion between multi-scale maps, and the s-scale feature map lacks information for small object detection because of the loss of spatial dimension information during the resizing process. On the other hand, for medium-sized feature maps, fusion between small- and large-sized feature maps is relatively sufficient. The medium-sized feature map that is finally output after fusion reflects high-level semantic information and low-level spatial information, so it will be useful on its own for small object detection tasks. Accordingly, we reduced the neck structure to use only the m-scale feature map for the real-time detection task, which ultimately made it possible to utilize feature information at various levels without incurring a large computational burden. Table 4 shows a comparison before and after application of Efficient-RepGFPN. Compared to the case for which the structure was not applied, recall increased by 1.2% and

mAP increased by up to 3.5%. And when comparing results by feature map sizes, it can be seen that the m-scale feature map resulted in the best performance.

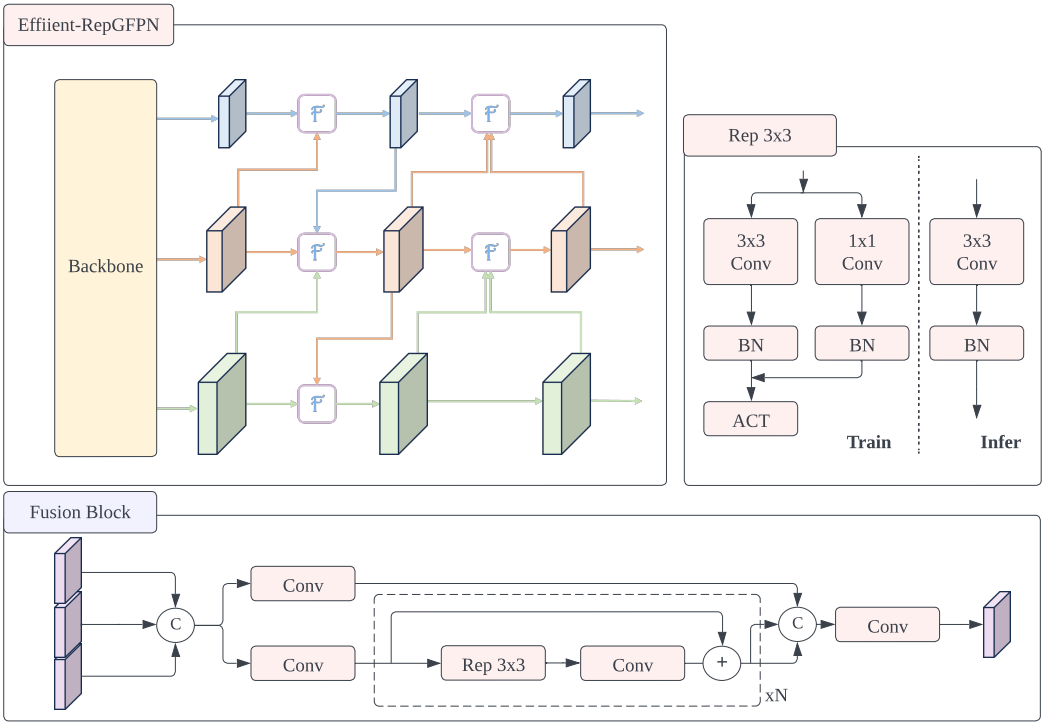



**Figure 5.** Overall structure of Efficient-RepGFPN.

**Table 4.** Ablation Table 3: DFConv + E-RGFPN_(s, m, l). Comparison of the detection performance by output feature map size of Efficient-RepGFPN to be used in the detection layer. DCEF$^2$-YOLO used the m-feature for detection because the detection rate, accuracy, and mAP were the highest with a medium-sized feature map.

| | Precision | Recall | mAP@0.5 | mAP@0.5:0.95 |
|---|---|---|---|---|
| w/o E-RGFPN | 0.662 | 0.622 | 0.648 | 0.372 |
| w/E-RGFPN-s | 0.610 | 0.520 | 0.668 | 0.385 |
| w/E-RGFPN-m | **0.709** | **0.634** | **0.683** | **0.403** |
| w/E-RGFPN-l | 0.667 | 0.532 | 0.630 | 0.365 |

### *3.3. Focus to Conv*

YOLOv5 uses the focus module to adjust the size of network input images. The focus module expands the image from the spatial dimension to the depth dimension and generates an image with an appropriate resolution and a deeper channel for convolution operations. In this process, the image is divided into grid units and is processed, which can cause information loss in small objects, similar to the patch embedding of the transformer mentioned above. In YOLOv6 [54], the focus module was replaced with convolution-based processing for faster computation on the GPU. According to [7], this change improves small object detection model performance because during the training process, the corresponding input image processing convolution operation is also trained to improve the detection performance. Figure 6 compares the focus module and the convolution-based input preprocessing stage, and Table 5 compares the performance according to whether or not the focus module was replaced. Considerable improvement in small object learning performance was achieved.

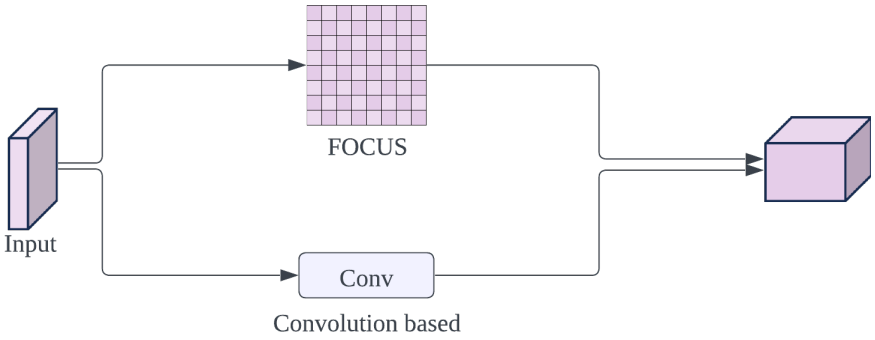



**Figure 6.** Input data processing comparison with Focus and Convolution.

**Table 5.** Ablation Table 4 (Input data processing). Input image processing through learning significantly improves the performance of small object detection tasks compared to existing input image processing.

| | Precision | Recall | mAP@0.5 | mAP@0.5:0.95 |
|---|---|---|---|---|
| Focus | 0.5065 | 0.622 | 0.599 | 0.324 |
| Conv | **0.662** | **0.622** | **0.648** | **0.372** |

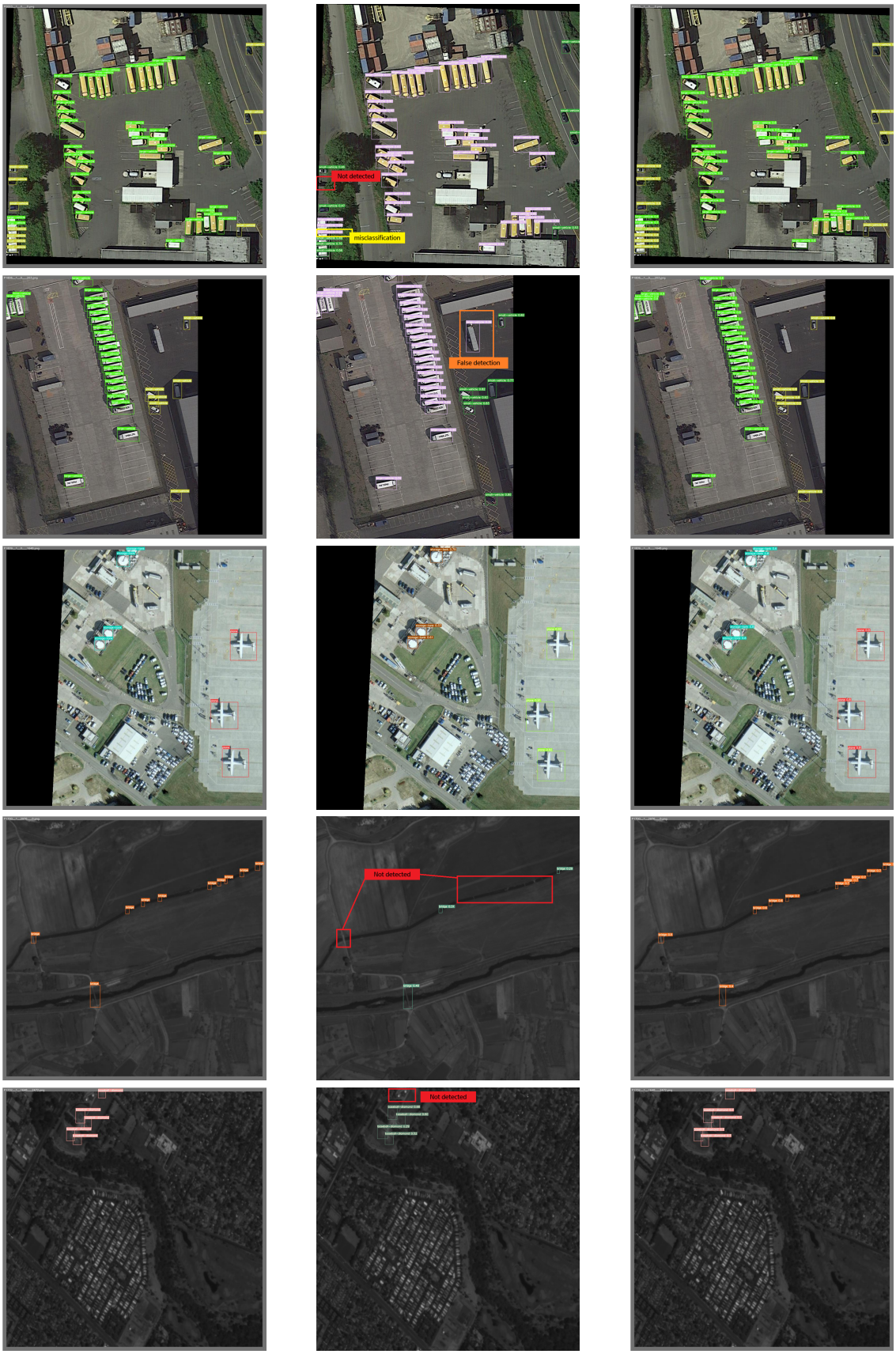
## 4. Experimental Results

In this study, we evaluated the comprehensive detection performance of DCEF$^2$-YOLO using DOTA and NWPU VHR-10, which are well-known aerial object detection benchmark datasets, as well as the VEDAI dataset, which specifically focuses on small objects. Our proposed model exhibited a mAP improvement of +6.1% compared to the baseline model on the DOTA-v1.0 test set, +1.5% on the VEDAI512 test set, and +0.3% on the NWPU VHR-10 test set. Using a GeFore RTX 3090, the network had a detection speed of 120.48 FPS on 512 × 512 images, confirming its suitability as a real-time small object detection model for aerial images. DOTA and VEDAI image results are compared with the results of training each aerial dataset on YOLOv7, which is widely recognized as a superior object detector in terms of speed and accuracy. Furthermore, through inference on the NWPU VHR-10 test set, we conducted a comparison between the detection results and the ground truth and analyzed the detection performance of the proposed network.

### *4.1. DOTA*

DOTA is a large-scale dataset for object detection in aerial images [10], consists of images collected by Google Earth, GF-2, and JL-1 satellites, and is provided by the Chinese Center for Resources for Satellite Data and Applications. The dataset contains objects of various sizes, orientations, and shapes and consists of RGB images and gray-scale images with various resolutions ranging from 800 × 800 to 20,000 × 20,000 pixels. DOTA provides oriented and horizontal bounding box annotation and is currently available in three versions: DOTA-v1.0, v1.5, and v2.0.

The proposed model was evaluated on the DOTA-v1.0 (HBB) dataset. For fair verification, Table 6 shows a comparison using the same settings as Tian et al. [39]. The DOTA dataset is utilized for training by dividing it into patches due to its irregular and very large image size. All images were divided into patches with a size of 1024 × 1024 and with an overlap range of 200 pixels, and 10,000 patches were randomly selected and divided at a ratio of 7:1:2 and used as training, validation, and test sets, respectively. Training lasted for 150 epochs. The dataset consisted of 15 classes: plane (PL), baseball diamond (BD), bridge (BR), ground track field (GTF), small vehicle (SV), large vehicle (LV), ship (SH), tennis court (TC), basketball court (BC), storage tank (ST), soccer ball field (SBF), roundabout (RA), harbor (HA), swimming pool (SP), and helicopter (HC). Figure 7 is an example of detection results for the DOTA test set.




(**a**) Ground truth (**b**) YOLOv7 (trained with DOTA) (**c**) Ours

**Figure 7.** Detection results comparison using the DOTA dataset. While YOLOv7 frequently exhibits false detections, non-detections, and misclassifications, the proposed network consistently delivers accurate detection results.

**Table 6.** DOTA-v1.0 test results (10,000 image patches selected randomly). This shows that the relatively low large object detection performance has been greatly complemented by small- and medium-sized object detection performance and proves that the proposed model can be effectively applied to small- and medium-sized object detection tasks and shows good performance.

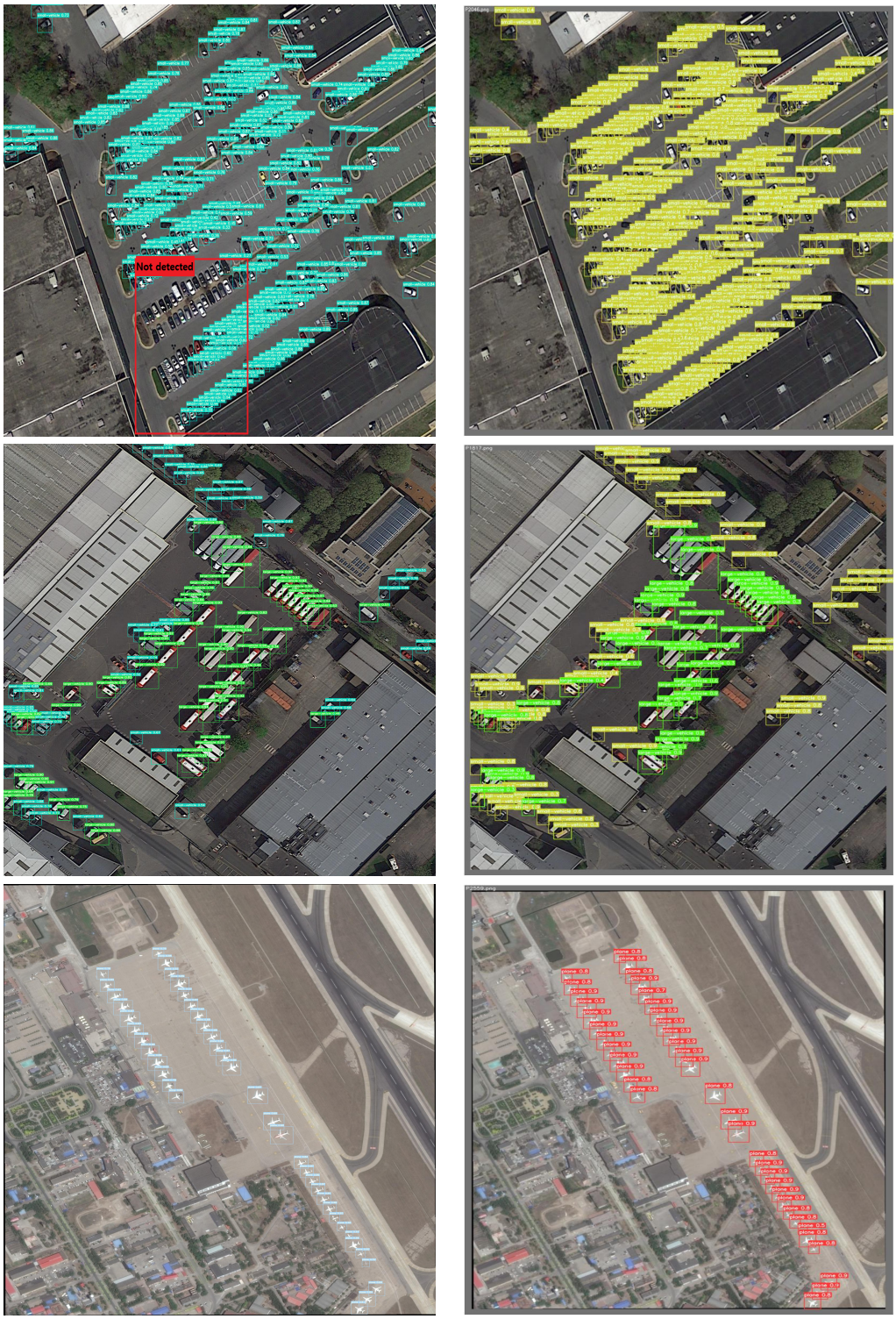
| Method | GFLOPs | mAP@0.5 | mAP@0.5:0.95 | $AP_S^{val}$ | $AP_M^{val}$ |
|---|---|---|---|---|---|
| YOLOv3 | 154.8 | 59.4 | 38.7 | 14.0 | 34.2 |
| YOLOv3-Spp | 156.6 | 60.3 | 39.0 | 15.6 | 34.3 |
| YOLOv4-Csp | 119.1 | 60.3 | 39.6 | 15.6 | 34.2 |
| YOLOR-Csp | 107.9 | 60.0 | 39.6 | 16.6 | 34.1 |
| YOLOv5l | 108.7 | 59.9 | 40.0 | 14.5 | 33.8 |
| YOLOv5-Bifpn | **15.7** | 58.1 | 36.4 | 12.8 | 31.4 |
| YOLOv5s-Transformer (7.0) | 119.1 | 61.8 | 40.4 | 15.1 | 34.7 |
| YOLOv6l | 156.6 | 60.0 | 40.1 | 15.8 | 35.8 |
| YOLOv7 | 103.4 | 60.0 | 40.0 | 15.9 | 34.4 |
| DCN-YOLO [39] | 98.4 | 63.4 | 41.9 | 20.1 | 36.4 |
| DCEF$^2$-YOLO | 96.4 | **69.5** | **45.8** | **23.6** | **46.6** |

Table 6 compares the performance of the proposed model with the SOTA model DCN-YOLO announced in July 2023 and other YOLO-based models. DCEF$^2$-YOLO showed improved performance by mAP@0.5 +6.1% and mAP@0.5:0.95 +3.9% with the lowest computational cost (from -2 GFLOPs up to -60.2 GFLOPs). This proves that DCEF$^2$-YOLO has successfully achieved low computation and high detection performance through appropriate optimization and combination of each structure and module. Furthermore, the verification results were compared according to the object size. Similar to the COCO benchmark method, the results were divided into small- and medium-sized objects of 32 × 32 pixels and 96 × 96 pixels, respectively, and AP values were calculated for each size. The proposed model, which focuses on improving the performance of small object detection in aerial images, showed significantly improved performance in $AP_S$ and $AP_M$, which are small and medium object detection results, respectively. This proves that the proposed model can be effectively applied to aerial detection tasks and shows good performance.

Table 7 compares the results of sufficient training for 300 epochs by class for all patches rather than 10,000 patches to verify the scalability of the proposed model. Compared to the training results of 150 epochs on 10,000 randomly selected patches in Table 6, mAP@0.5 showed an improvement of +14.2% to 83.7mAP@0.5. This is similar performance to the DOTA-v1.0 benchmark model. Therefore, DCEF$^2$-YOLO can bring about a clear performance improvement after sufficient training with a sufficient dataset. And for inference purposes, Figure 8 is an example of detection results for the entire image rather than a patch image. It can be observed that the proposed network consistently delivers more stable detection results compared to YOLOv7.

**Table 7.** DOTA-v1.0 test results (all image patches). The detection performance will be improved if sufficient training is performed with sufficient data.

| | All (%) | PL | BD | BR | GTF | SV | LV | SH |
|---|---|---|---|---|---|---|---|---|
| Precision | 89.7 | 97.9 | 91.1 | 82.3 | 91.0 | 89.5 | 94.9 | 95.5 |
| Recall | 80.2 | 86.7 | 82.0 | 80.6 | 47.9 | 85.8 | 94.0 | 90.5 |
| mAP@0.5 | 83.7 | 92.7 | 86.2 | 82.8 | 54.2 | 86.5 | 95.8 | 91.2 |
| mAP@0.5:0.95 | 60.9 | 73.9 | 58.7 | 50.0 | 33.2 | 62.8 | 79.5 | 73.5 |
| | **TC** | **BC** | **ST** | **SBF** | **RA** | **HA** | **SP** | **HC** |
| Precision | 98.1 | 92.9 | 91.5 | 65.5 | 84.4 | 90.6 | 85.6 | 94.9 |
| Recall | 98.6 | 90.5 | 90.9 | 31.6 | 79.6 | 76.3 | 89.0 | 79.5 |
| mAP@0.5 | 99.4 | 92.4 | 92.6 | 40.8 | 82.6 | 84.2 | 90.4 | 84.0 |
| mAP@0.5:0.95 | 92.5 | 79.3 | 67.1 | 20.9 | 53.5 | 57.4 | 51.4 | 59.8 |




(**a**) YOLOv7 (trained with DOTA) (**b**) Ours

**Figure 8.** Detection results comparison on whole images in the DOTA test set for inference. These are comparison of detection results on entire images, not on the patch-wise images used for training. It can be observed that the proposed network consistently delivers more stable detection results compared to YOLOv7.

### *4.2. VEDAI*

VEDAI is a dataset for vehicle detection in aerial images [32] and provides RGB and IR image sets that are 512 × 512 and 1024 × 1024 in size. Each consists of approximately 1200 images, and a 512 × 512 RGB set was used to verify the performance of the proposed model. Each image has a spatial resolution of 25 cm at 512 × 512 and contains 3757 small instances from 8 to 20 pixels in width. The dataset was divided at a ratio of 8:1:1 for training, validation, and test sets, respectively. The dataset includes nine classes: car, truck, pickup, tractor, camper, boat, van, plane, and other instances. Table 8 compares the performance with YOLO-based models, and Table 9 compares the detection performance by class with [6,7,55], which were also proposed for aerial object detection.

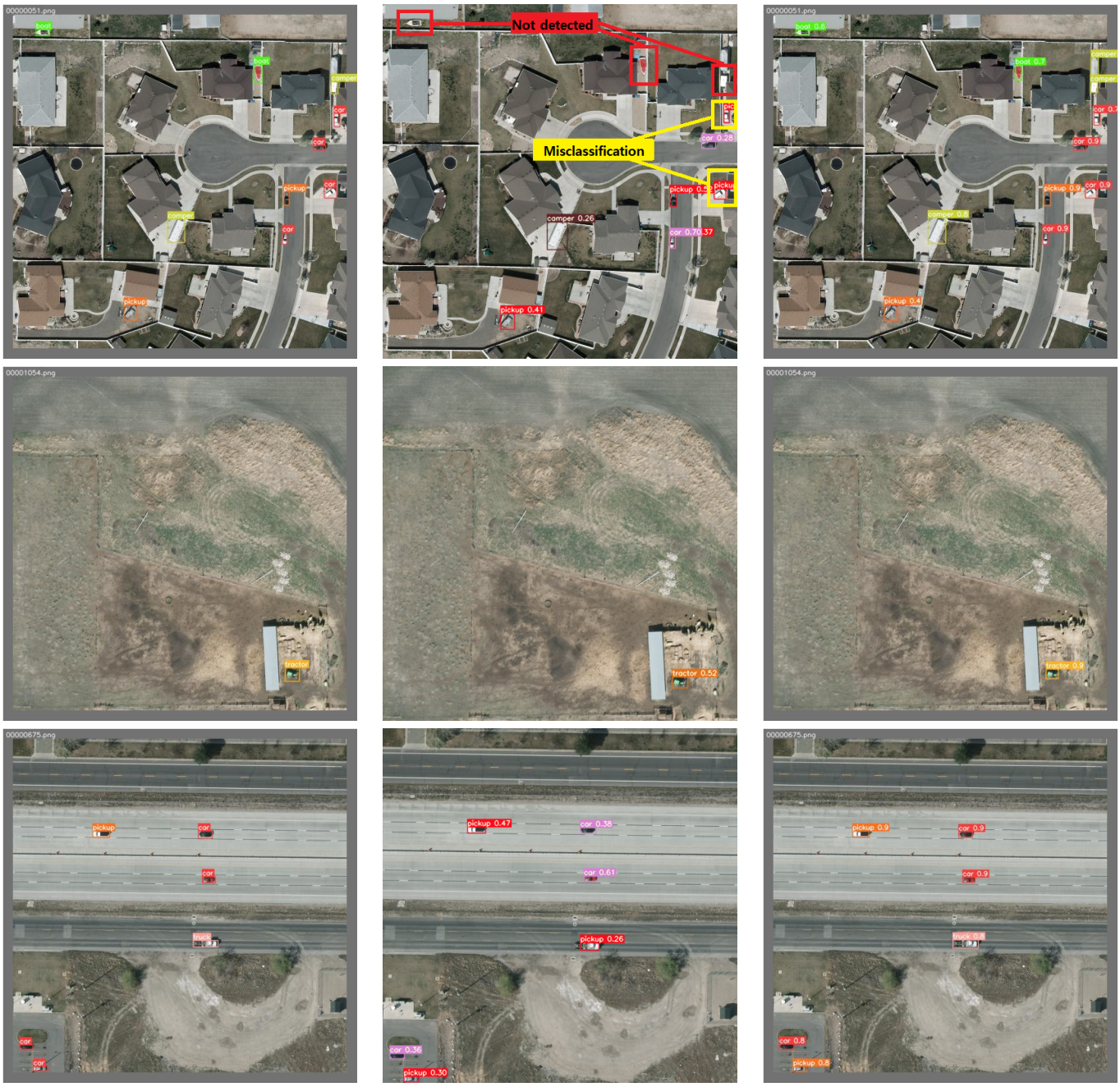
**Table 8.** VEDAI512 test results.

| Method | Precision | Recall | F1 Score | mAP@0.5 | mAP@0.5:0.95 | GFLOPs |
|---|---|---|---|---|---|---|
| YOLOv5l | 50.3 | **64.4** | 56.5 | 52.5 | 27.6 | 115.6 |
| YOLOv5x | 64.0 | 52.7 | 57.8 | 55.9 | 30.8 | 219.0 |
| YOLOR-Csp | 20.2 | 47.2 | 28.3 | 31.4 | 19.5 | 120.6 |
| YOLOR-Csp*x* | 23.9 | 56.1 | 33.5 | 34.7 | 21.1 | 222.4 |
| YOLOX-l | 28.7 | 49.1 | 36.2 | 55.6 | 28.8 | 155.6 |
| YOLOX-x | 21.2 | 59.9 | 31.3 | 53.4 | 26.6 | 281.9 |
| ECAPs-YOLOv5l | 64.6 | 51.4 | 57.2 | 56.0 | 31.5 | 122.1 |
| ECAP-YOLOv5l | 71.6 | 52.9 | 60.8 | 58.7 | 34.5 | 132.4 |
| YOLOv7 | - | - | - | 61.1 | - | - |
| SuperYOLO | - | - | - | 63.1 | - | - |
| BiCAM-Detector [55] | - | - | - | 63.8 | - | - |
| DCEF$^2$-YOLO | **74.0** | 60.0 | **66.3** | **65.3** | **37.2** | **96.4** |
| DCEF$^2$-YOLO (500ep val) | **72.1** | **67.2** | **69.6** | **68.6** | **42.3** | **96.4** |

**Table 9.** VEDAI512 test results per class.

| Method | All(%) | Cars | Trucks | Pickups | Tractors | Campers | Boats | Vans | Planes | Others |
|---|---|---|---|---|---|---|---|---|---|---|
| avg. pixels | | 237 | 546 | 309 | 936 | 323 | 308 | 416 | - | 588 |
| YOLOv5l | 52.5 | 79.5 | 43.3 | 66.2 | 63.2 | 53.3 | 26.4 | 45.1 | 63.4 | 32.1 |
| YOLOv5x | 55.9 | 83.6 | 59.4 | 67.6 | 62.8 | 49.8 | 24.3 | 53.5 | 61.5 | **40.7** |
| ECAPs-YOLOv5l | 56.0 | 85.5 | 40.4 | 74.8 | 61.8 | 54.7 | 32.7 | 39.8 | 90.1 | 23.9 |
| ECAP-YOLOv5l | 58.7 | 85.6 | 52.0 | 77.0 | 62.7 | 50.3 | 35.0 | 47.8 | 91.0 | 27.2 |
| YOLOv7 | 61.1 | 72.6 | **59.5** | 48.6 | 60.1 | 69.3 | 57.8 | 21.7 | 98.7 | - |
| SuperYOLO | 63.1 | 76.9 | 51.5 | 56.8 | 59.7 | 74.8 | 52.9 | 34.6 | 97.5 | - |
| BiCAM-Detector | 63.8 | 82.1 | 44.5 | 58.2 | 57.4 | **82.6** | **55.1** | 31.3 | **99.5** | - |
| **DCEF$^2$-YOLO** | **65.3** | **92.2** | 52.2 | **81.2** | **80.3** | 62.8 | 49.4 | **72.7** | 63.6 | 33.3 |

In Table 8, DCEF$^2$-YOLO showed the best balance between detection rate and accuracy (F1 score) and the highest detection rate and mAP compared to the comparison models. In particular, compared to YOLOv7, there was a 12.1% improvement in detection rate and a 25.4% improvement in mAP. Additionally, when compared to the model proposed for small object detection [55] in June 2023, there was a mAP improvement of 1.5%. And the computational amount was also 36 GFLOPs lower than the YOLOv5-based small object detection comparison model and 7.1 GFLOPs lower than YOLOv7. YOLOv7 has superior processing speed among object detectors, and with the VEDAI dataset, it has a processing speed of 88.49 FPS. However, DCEF$^2$-YOLO has a processing speed of 120.48 FPS. This means that 31.99 more images per second can be processed with the proposed method with high performance even with higher efficiency. Figure 9 is an comparison of detection results on VEDAI test set. By comparing the results with YOLOv7, it is evident that the proposed network exhibits significantly more stable and accurate detection of small objects.




(**a**) Ground Truth (**b**) YOLOv7 (trained with VEDAI) (**c**) Ours

**Figure 9.** Comparison of detection results with VEDAI. YOLOv7 frequently causes non-detection and misclassification in complex images, while the proposed network outputs more appropriate detection results. Additionally, YOLOv7 has an overall lower score for each detected object than the proposed network, making it less reliable. This proves that the proposed network is better optimized for small object detection tasks than YOLOv7, which was proposed for a variety of tasks.

### *4.3. NWPU VHR-10*

The NWPU VHR-10 dataset [37] comprises 800 high-resolution remote sensing images collected from Google Earth and Vaihingen that have been annotated manually by experts. It provides 650 labeled images and 150 negative images without objects, with images having widths ranging from around 400 to 1000 pixels. It contains 3775 instances with widths and heights ranging from around 20 to 400 pixels and primarily consisting of medium-sized objects. We utilized NWPU VHR-10 for experimentation to verify the scalability of the network and for comprehensive performance analysis. We split the 650 labeled images into

a ratio of 8:2 for training and testing, respectively. The dataset consists of 10 classes: airplane (AP), ship (SH), storage tank (ST), baseball diamond (BD), tennis court (TC), basketball court (BC), ground track field (GT), harbor (HA), bridge (BR), and vehicle (VE).

The proposed network achieved a precision of 90.7% and a mAP@0.5 of 94.7% at a recall of 93.1%. Furthermore, even with a relatively lower recall of approximately 89.3%, the mAP@0.5 increased to 95.7%. We compared the performance of the first training result, which achieved sufficiently high recall, with recent aerial image-based detection networks. Table 10 compares the detection performance on the NWPU VHR-10 test set. The proposed network exhibited a mAP 0.3% higher than the top-performing MSCCA [56] among the comparison networks, +0.6% compared to DS-YOLOv8 [57], which was proposed in 2023, and +1.3% compared to SuperYOLO [7], which was proposed in 2022.

**Table 10.** Comparison of NWPU VHR-10 test results.

| Method | mAP@0.5 |
|---|---|
| MSF-SNET [58] | 82.4 |
| CBFF-SSD [59] | 91.4 |
| SuperYOLO [7] | 93.3 |
| DS-YOLOv8 [57] | 94.1 |
| MSCCA [56] | 94.4 |
| DCEF$^2$-YOLO | **94.7** |

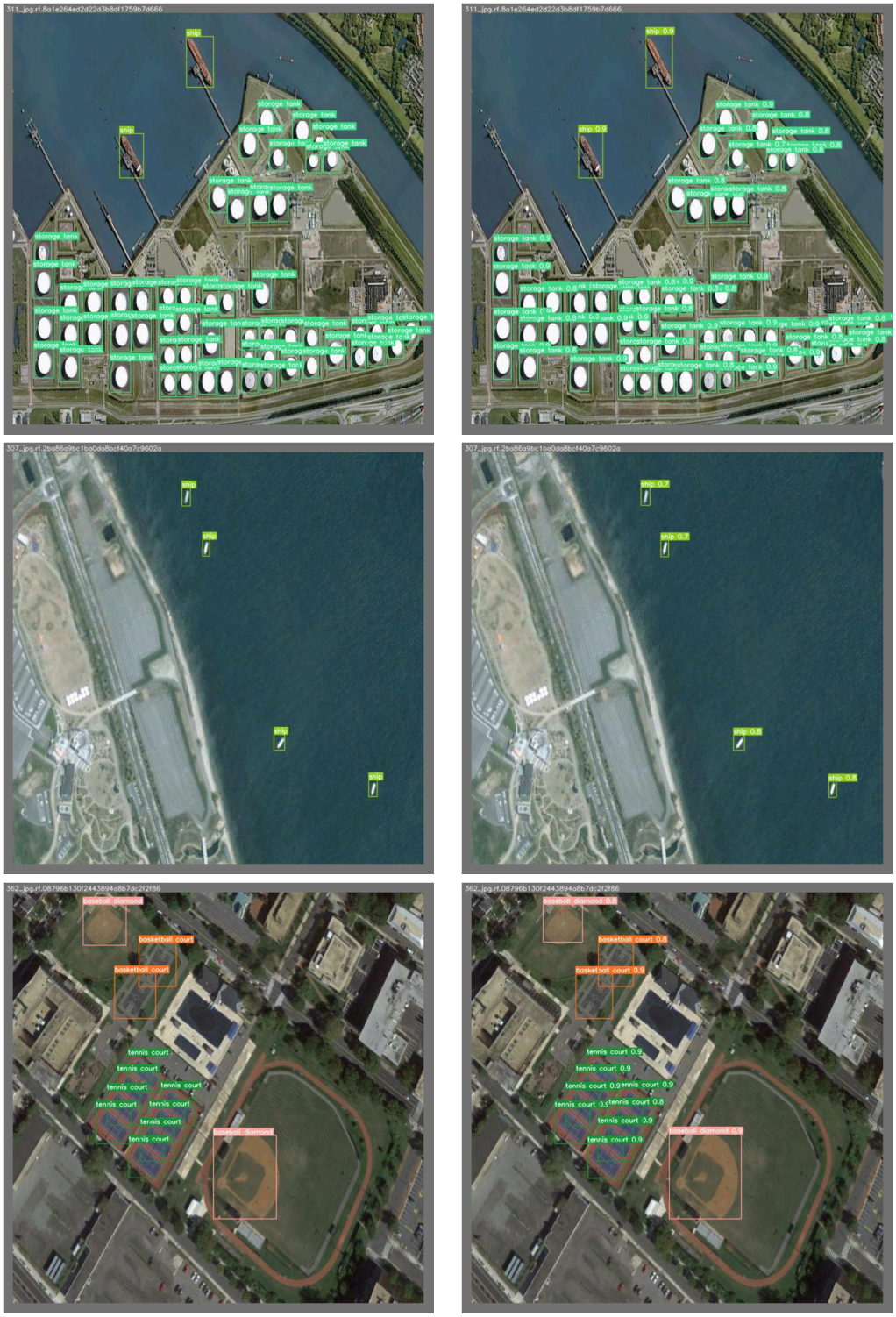
Table 11 compares the detection performance class-wise. The highest values are highlighted in bold, and the second-highest values are underlined. Despite most objects being medium-sized or larger, the proposed network consistently exhibits high detection performance among the compared networks, achieving mAPs exceeding 90% for all classes except VE.

**Table 11.** NWPU VHR-10 test results per class. The detection performance for NWPU VHR-10 is compared class-wise. The highest values are emphasized in bold, and the second-highest values are underlined. It can be observed that the proposed network consistently demonstrates high detection performance among compared networks.

| Method | All (%) | AP | SH | ST | BD | TC | BC | GT | HA | BR | VE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| MSF-SNET | 82.4 | 93.5 | 92.2 | 58.8 | <u>97.9</u> | 65.1 | 79.5 | 94.7 | 75.6 | <u>91.4</u> | 75.5 |
| CBFF-SSD | 91.4 | 96.9 | **94.3** | 81.0 | **99.1** | <u>91.5</u> | 92.6 | **98.8** | <u>91.6</u> | 89.7 | 78.8 |
| MSCCA | <u>94.4</u> | **99.7** | 90.4 | <u>90.8</u> | 90.8 | 90.8 | **98.6** | <u>98.3</u> | 90.3 | 88.2 | **98.3** |
| **DCEF$^2$-YOLO** | **94.7** | <u>99.6</u> | <u>92.9</u> | **97.3** | 95.6 | **97.3** | <u>98.4</u> | 93.9 | **93.4** | **92.1** | <u>86.7</u> |

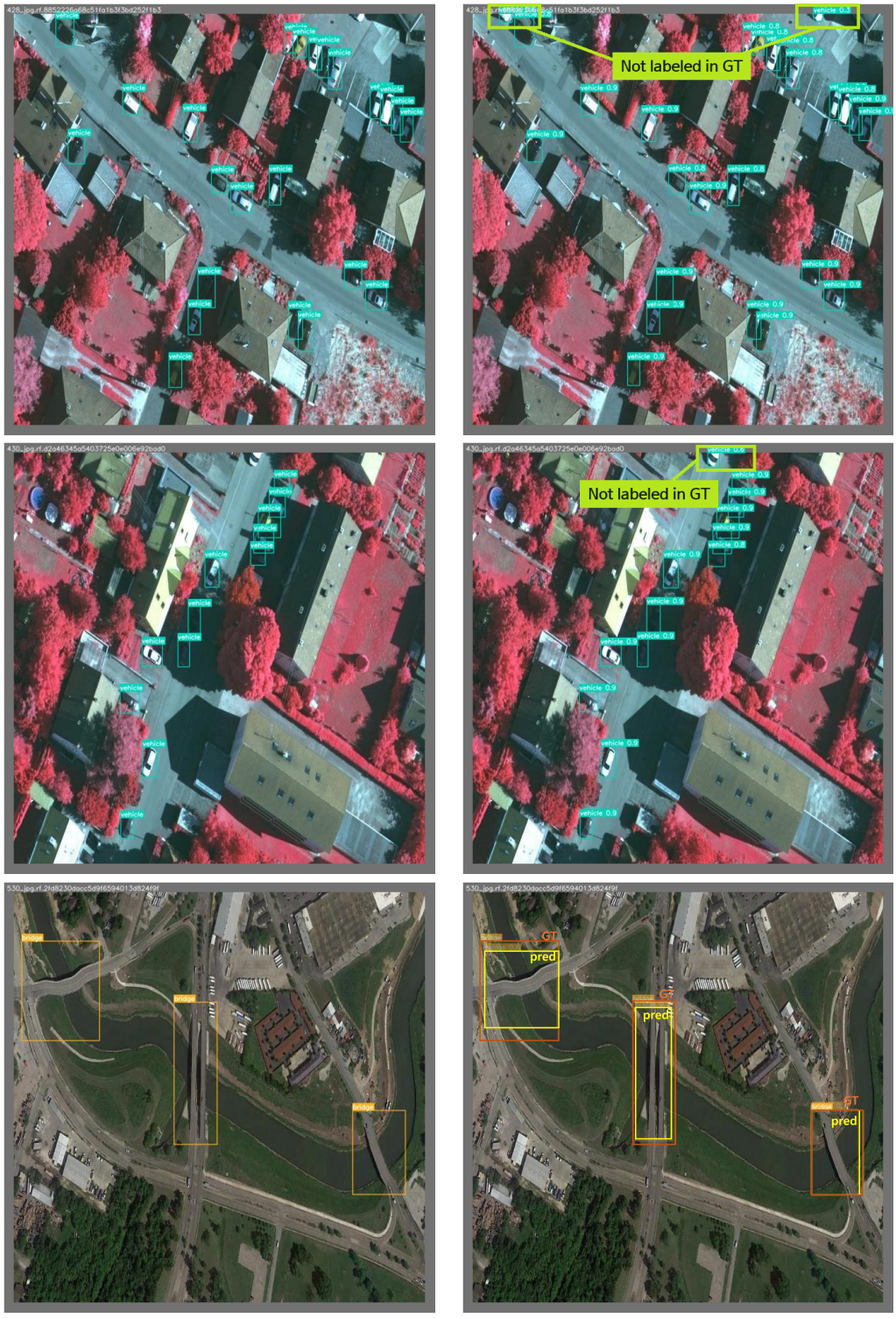
Figure 10 shows the inference images for NWPU VHR-10, while Figure 11 specifically focuses on the inference images for the VE class. Here, we were able to identify the reason for the relatively low mAP of the proposed network for VE through the inference images.

Figure 11 depicts the detection of unlabeled vehicles that are partially visible in the image. This indicates that objects were detected even when only a portion of them was visible, showcasing the network's effective utilization of internal object information. Such instances were particularly common, especially in the VE class. In other classes, it was observed that the proposed network often produced bounding boxes that better fit the objects compared to the ground truth bounding boxes. This issue stems from the limitations of manually labeled datasets and potentially impacts numerical detection performance metrics like recall and mAP. However, these results ultimately affirm the proposed network's capability to accurately detect objects even in situations where feature information is lacking.



(**a**) Ground Truth (**b**) Ours

**Figure 10.** Detection results with the NWPU VHR-10 test set.




(**a**) Ground Truth (**b**) Ours

**Figure 11.** Some comparison examples of detection results. When comparing the ground truth (GT) images on the left and the results on the right in rows 1 and 2, it is observed that objects that were partially visible and thus not labeled previously were detected. In row 3, it can be seen that the proposed network frequently outputs bounding boxes that better fit the shape of objects compared to the GT bounding boxes. While this may have a negative impact on numerical performance metrics, it ultimately demonstrates the proposed network's capability to accurately detect objects even in scenarios where feature information is lacking.

## 5. Discussion and Conclusions

This paper proposed DCEF$^2$-YOLO for effective real-time small object detection in aerial images. The DFConv module was appropriately applied to learn small objects with fewer pixels and low resolution, effectively enabling intensive learning of small object areas. The Efficient-RepGFPN neck is optimized to output only medium-sized feature maps, ultimately enabling 'real-time' detection. This also provides high-quality feature maps to the detection layer, effectively improving performance. The convolution-based input stage also contributes to faster computation and improved performance. Thus, several techniques suitable for 'real-time' 'small' object detection tasks are optimized and applied appropriately to DCEF$^2$-YOLO so that we successfully achieved our goal. DCEF$^2$-YOLO showed good performance at the SOTA level on aerial object detection benchmark sets such as DOTA, VEDAI, and NWPU VHR-10 and showed the possibility of expansion to various detection tasks. DCEF$^2$-YOLO is expected to be applied to embedded systems and to be effectively utilized in various industrial environments, such as real-time traffic monitoring and military reconnaissance.

**Author Contributions:** Conceptualization, Y.S.; methodology, Y.S.; software, Y.S.; validation, Y.S.; formal analysis, Y.S.; investigation, Y.S.; data curation, Y.S.; writing—original draft preparation, Y.S.; writing—review and editing, Y.S.; visualization, Y.S.; supervision, S.K. and H.S.; project administration, S.K., H.S., J.O., M.B. and J.Y.; funding acquisition, H.S., J.O., M.B. and J.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by Korea Research Institute for defense Technology planning and advancement (KRIT) - Grant funded by Defense Acquisition Program Administration (DAPA) (KRIT-CT-22-060).

**Data Availability Statement:** Vehicle detection in aerial imagery (VEDAI) dataset (https://downloads.greyc.fr/vedai/, accessed on 17 March 2024), DOTA dataset (https://captain-whu.github.io/DOTA/dataset.html, accessed on 17 March 2024), NWPU VHR-10 dataset (https://github.com/chaozhong2010/VHR-10_dataset_coco, accessed on 17 March 2024).

**Conflicts of Interest:** Authors Heesub Shin, Jaewoo Ok, Minyoung Back, Jaehyuk Youn were employed by the company LIG Nex1. 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. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
2. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. *arXiv* **2014**, arXiv:1405.0312.
3. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 2012. Available online: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (accessed on 17 March 2024).
4. Zhou, Z.; Siddiquee, M.M.R.; Tajbakhsh, N.; Liang, J. UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation. *IEEE Trans. Med. Imaging* **2019**, *39*, 1856–1867. [CrossRef]
5. Xiang, T.; Zhang, C.; Liu, D.; Song, Y.; Huang, H.; Cai, W. BiO-Net: Learning recurrent bi-directional connections for encoder-decoder architecture. In Proceedings of the Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, 4–8 October 2020; pp. 74–84.
6. Kim, M.; Jeong, J.; Kim, S. ECAP-YOLO: Efficient channel attention pyramid YOLO for small object detection in aerial image. *Remote Sens.* **2021**, *13*, 4851. [CrossRef]
7. Zhang, J.; Lei, J.; Xie, W.; Fang, Z.; Li, Y.; Du, Q. SuperYOLO: Super resolution assisted object detection in multimodal remote sensing imagery. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–15. [CrossRef]
8. Wang, Y.; Wang, C.; Zhang, H.; Dong, Y.; Wei, S. A SAR Dataset of Ship Detection for Deep Learning under Complex Backgrounds. *Remote Sens.* **2019**, *11*, 765. [CrossRef]
9. Long, Y.; Jiang, X.; Liu, X.; Zhang, Y. Sar Atr with Rotated Region Based on Convolution Neural Network. In Proceedings of the IGARSS 2019—2019 IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan, 28 July 2019–2 August 2019; pp. 1184–1187. [CrossRef]

10. 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 (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
11. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 10012–10022.
12. 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.
13. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. Available online: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html (accessed on 17 March 2024).
14. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16 × 16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
15. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
16. 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.
17. Zhu, X.; Hu, H.; Lin, S.; Dai, J. Deformable convnets v2: More deformable, better results. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9308–9316.
18. Zhong, B.; Yang, L. Improved YOLOv5 in Remote Sensing Slender and Rotating Target Detection. In Proceedings of the 2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA), Changchun, China, 25–27 February 2022; pp. 918–923.
19. Wang, L.; Cao, Y.; Wang, S.; Song, X.; Zhang, S.; Zhang, J.; Niu, J. Investigation into recognition algorithm of helmet violation based on YOLOv5-CBAM-DCN. *IEEE Access* **2022**, *10*, 60622–60632. [CrossRef]
20. Liu, Y.; He, G.; Wang, Z.; Li, W.; Huang, H. NRT-YOLO: Improved YOLOv5 based on nested residual transformer for tiny remote sensing object detection. *Sensors* **2022**, *22*, 4953. [CrossRef]
21. Jocher, G. Ultralytics YOLOv5. 2020. Available online: https://doi.org/10.5281/zenodo.3908559 (accessed on 17 March 2024).
22. Xu, X.; Jiang, Y.; Chen, W.; Huang, Y.; Zhang, Y.; Sun, X. Damo-yolo: A report on real-time object detection design. *arXiv* **2022**, arXiv:2211.15444.
23. Pu, Y.; Wang, Y.; Xia, Z.; Han, Y.; Wang, Y.; Gan, W.; Wang, Z.; Song, S.; Huang, G. Adaptive Rotated Convolution for Rotated Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–3 October 2023.
24. Yi, J.; Wu, P.; Liu, B.; Huang, Q.; Qu, H.; Metaxas, D. Oriented object detection in aerial images with box boundary-aware vectors. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Online, 5–9 January 2021; pp. 2150–2159.
25. Yang, X.; Zhou, Y.; Zhang, G.; Yang, J.; Wang, W.; Yan, J.; Zhang, X.; Tian, Q. The KFIoU loss for rotated object detection. *arXiv* **2022**, arXiv:2201.12558.
26. Wang, X.; Wang, G.; Dang, Q.; Liu, Y.; Hu, X.; Yu, D. PP-YOLOE-R: An Efficient Anchor-Free Rotated Object Detector. *arXiv* **2022**, arXiv:2211.02386.
27. Fang, Y.; Wang, Z. Cross-modality attentive feature fusion for object detection in multispectral remote sensing imagery. *Pattern Recognit.* **2022**, *130*, 108786.
28. Li, Y.; Hou, Q.; Zheng, Z.; Cheng, M.M.; Yang, J.; Li, X. Large Selective Kernel Network for Remote Sensing Object Detection. *arXiv* **2023**, arXiv:2303.09030.
29. Wang, D.; Zhang, J.; Du, B.; Xia, G.S.; Tao, D. An Empirical Study of Remote Sensing Pretraining. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–20. [CrossRef]
30. Liu, Z.; Wang, H.; Weng, L.; Yang, Y. Ship Rotated Bounding Box Space for Ship Extraction From High-Resolution Optical Satellite Images With Complex Backgrounds. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 1074–1078. [CrossRef]
31. Cheng, G.; Wang, J.; Li, K.; Xie, X.; Lang, C.; Yao, Y.; Han, J. Anchor-Free Oriented Proposal Generator for Object Detection. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–11. [CrossRef]
32. Razakarivony, S.; Jurie, F. Vehicle detection in aerial imagery: A small target detection benchmark. *J. Vis. Commun. Image Represent.* **2016**, *34*, 187–203. [CrossRef]
33. Xia, G.; Hu, J.; Hu, F.; Shi, B.; Bai, X.; Zhong, Y.; Zhang, L. AID: A Benchmark Dataset for Performance Evaluation of Aerial Scene Classification. *arXiv* **2016**, arXiv:1608.05167.
34. Lam, D.; Kuzma, R.; McGee, K.; Dooley, S.; Laielli, M.; Klaric, M.; Bulatov, Y.; McCord, B. xView: Objects in Context in Overhead Imagery. *arXiv* **2018**, arXiv:1802.07856.
35. Waqas Zamir, S.; Arora, A.; Gupta, A.; Khan, S.; Sun, G.; Shahbaz Khan, F.; Zhu, F.; Shao, L.; Xia, G.S.; Bai, X. iSAID: A Large-scale Dataset for Instance Segmentation in Aerial Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–17 June 2019; pp. 28–37.

36. Boguszewski, A.; Batorski, D.; Ziemba-Jankowska, N.; Dziedzic, T.; Zambrzycka, A. LandCover.ai: Dataset for Automatic Mapping of Buildings, Woodlands, Water and Roads from Aerial Imagery. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Nashville, TN, USA, 20–25 June 2021; pp. 1102–1110.
37. Cheng, G.; Han, J.; Zhou, P.; Guo, L. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS J. Photogramm. Remote Sens.* **2014**, *98*, 119–132. [CrossRef]
38. Zhu, P.; Wen, L.; Du, D.; Bian, X.; Fan, H.; Hu, Q.; Ling, H. Detection and tracking meet drones challenge. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 7380–7399. [CrossRef] [PubMed]
39. Tian, B.; Chen, H. Remote Sensing Image Target Detection Method Based on Refined Feature Extraction. *Appl. Sci.* **2023**, *13*, 8694. [CrossRef]
40. Wang, X.; Wang, A.; Yi, J.; Song, Y.; Chehri, A. Small Object Detection Based on Deep Learning for Remote Sensing: A Comprehensive Review. *Remote Sens.* **2023**, *15*, 3265. [CrossRef]
41. Redmon, J.; Divvala, S.K.; Girshick, R.B.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *arXiv* **2015**, arXiv:1506.02640.
42. Ren, S.; He, K.; Girshick, R.B.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**, arXiv:1506.01497.
43. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R.B. Mask R-CNN. *arXiv* **2017**, arXiv:1703.06870.
44. Bochkovskiy, A.; Wang, C.; Liao, H.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
45. Wang, C.; Liao, H.M.; Yeh, I.; Wu, Y.; Chen, P.; Hsieh, J. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *arXiv* **2019**, arXiv:1911.11929.
46. Jiang, Y.; Tan, Z.; Wang, J.; Sun, X.; Lin, M.; Li, H. GiraffeDet: A heavy-neck paradigm for object detection. *arXiv* **2022**, arXiv:2202.04256.
47. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
48. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. *arXiv* **2018**, arXiv:1803.01534.
49. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. *arXiv* **2019**, arXiv:1911.09070.
50. 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.
51. Ding, X.; Zhang, X.; Han, J.; Ding, G. Diverse Branch Block: Building a Convolution as an Inception-like Unit. *arXiv* **2021**, arXiv:2103.13425.
52. Ding, X.; Zhang, X.; Ma, N.; Han, J.; Ding, G.; Sun, J. Repvgg: Making vgg-style convnets great again. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13733–13742.
53. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H. Designing network design strategies through gradient path analysis. *arXiv* **2022**, arXiv:2211.04800.
54. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A single-stage object detection framework for industrial applications. *arXiv* **2022**, arXiv:2209.02976.
55. Xu, D.; Wu, Y. An Efficient Detector with Auxiliary Network for Remote Sensing Object Detection. *Electronics* **2023**, *12*, 4448. [CrossRef]
56. Ran, Q.; Wang, Q.; Zhao, B.; Wu, Y.; Pu, S.; Li, Z. Lightweight oriented object detection using multiscale context and enhanced channel attention in remote sensing images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 5786–5795. [CrossRef]
57. Shen, L.; Lang, B.; Song, Z. DS-YOLOv8-Based Object Detection Method for Remote Sensing Images. *IEEE Access* **2023**, *11*, 125122–125137. [CrossRef]
58. Lang, H.; Bai, Y.; Li, Y.; Jiang, D.; Zhang, Y.; Zhou, Q.; Wei, J.; Liu, J.; Zhang, Y.; Cui, T. A lightweight object detection framework for remote sensing images. *Remote Sens.* **2021**, *13*, 683.
59. Li, L.; Zhang, S.; Wu, J. Efficient object detection framework and hardware architecture for remote sensing images. *Remote Sens.* **2019**, *11*, 2376. [CrossRef]