applied sciences

MDPI

Article

A Dense Feature Pyramid Network for Remote Sensing Object Detection

Yu Sun [1], Wenkai Liu [1], Yangte Gao [2,3], Xinghai Hou [1] and Fukun Bi [1,*]

[1] Electronics and Communications Engineering, North China University of Technology, Beijing 100144, China; yusun@mail.ncut.edu.cn (Y.S.); liuwk@ncut.edu.cn (W.L.); 2019312100105@mail.ncut.edu.cn (X.H.)
[2] School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China; gaoyangte@qxslab.cn
[3] Qian Xuesen Laboratory of Space Technology, China Academy of Space Technology, Beijing 100094, China
* Correspondence: bifukun@ncut.edu.cn

Abstract: In recent years, object detection in remote sensing images has become a popular topic in computer vision research. However, there are various problems in remote sensing object detection, such as complex scenes, small objects in large fields of view, and multi-scale object in different categories. To address these issues, we propose DFPN-YOLO, a dense feature pyramid network for remote sensing object detection. To address difficulties in detecting small objects in large scenes, we add a larger detection layer on top of the three detection layers of YOLOv3, and we propose Dense-FPN, a dense feature pyramid network structure that enables all four detection layers to combine semantic information before sampling and after sampling to improve the performance of object detection at different scales. In addition, we add an attention module in the residual blocks of the backbone to allow the network to quickly extract key feature information in complex scenes. The results show that the mean average precision (mAP) of our method on the RSOD datasets reached 92%, which is 8% higher than the mAP of YOLOv3, and the mAP increased from 62.41% on YOLOv3 to 69.33% with our method on the DIOR datasets, outperforming even YOLOv4.

Keywords: remote sensing object detection; dense feature pyramid network; attention module; improved residual block; YOLO

Citation: Sun, Y.; Liu, W.; Gao, Y.; Hou, X.; Bi, F. A Dense Feature Pyramid Network for Remote Sensing Object Detection. *Appl. Sci.* **2022**, *12*, 4997. https://doi.org/10.3390/app12104997

Academic Editors: Qizhi Xu, Jin Zheng and Feng Gao

Received: 5 March 2022
Accepted: 13 May 2022
Published: 15 May 2022

1. Introduction

In recent years, with the development of machine learning and deep learning, object detection, which can be used in navigation [1], disaster warning [2], building detection [3], and other fields, has gradually become a popular research topic in computer vision. Object detection requires identifying and locating a specific object, such as an aircraft, a car, a pedestrian or another object, in an image scene. Object detection is a fundamental problem in the field of computer vision, along with typical tasks such as image classification [4], image segmentation [5], motion estimation [6], and object tracking [7], and it has prompted the development of a number of classical algorithms. However, it is still a difficult task to make machines learn to detect objects in remote sensing images [8], which have the problems of complex scenes, large scenes but small objects, and multi-scale objects [9] in different categories, and these make remote sensing object detection suffer from the problems of difficult detection of small objects and low accuracy of multiscale objects.

Traditional object detection method, such as the deformable parts model (DPM) [10,11], the histogram of oriented gradients [12]-support vector machine [13] (HOG-SVM), and the HOG-Cascade [14], are not ideal when applied directly to remote sensing object detection. Although these methods perform better when detecting common objects such as pedestrians and vehicles, because remote sensing images have complex backgrounds, large-scale differences of objects, and small objects, traditional detection algorithms are ineffective when detecting remote sensing objects. With the rapid development of computer

technology and deep learning, researchers have applied convolutional neural networks (CNNs) [15] to remote sensing object detection and achieved good results. J Redmon et al. proposed YOLOv3, an incremental improvement [16] over previous detection methods. Z Cui et al. proposed dense attention pyramid networks for multiscale ship detection in SAR images [17]. W Huang et al. proposed CF2PN [18], a cross-scale feature fusion pyramid network-based method for remote sensing object detection. D Xu et al. proposed FE-YOLO [19], a feature-enhancement network for remote sensing object detection. Compared with traditional object detection algorithms, object detection algorithms based on CNNs are more accurate, allowing them to detect multiscale objects and small objects in remote sensing images with high accuracy.

CNNs can extract spatial context information and have been widely used to detect objects in remote sensing images. At present, the most common neural networks for object detection are neural networks based on region proposal and neural networks based on anchor box regression. Most region proposal-based neural networks are two-stage networks that first determine the approximate object location based on the region proposal network and then accurately predict the object class and regress to the exact bounding box. While this step-by-step learning strategy improves the detection accuracy of these networks, it also increases the detection time and the difficulty in achieving efficient processing, and the training time is too long for remote sensing images with large input image sizes. Some typical examples of such networks include R-CNN [20], Fast R-CNN [21], and Faster R-CNN [22]. Most neural networks based on anchor box regression are one-stage networks that treat the whole prediction process as a regression process. This simplification of the process not only maintains the accuracy but also increases the speed; examples of such networks include the SSD [23–25], YOLO [26–28] series, and Efficientdet [29,30]. Among them, YOLO series networks are typical neural networks based on anchor box regression, and several versions, such as YOLOv2 [31], YOLOv3 [16], YOLOv4 [32], and YOLOv5 [33], have been open-sourced. Among these versions, YOLOv3, YOLOv4, and YOLOv5 achieve a good balance between speed and accuracy when faced with the demands of traditional object detection applications, and they can achieve both efficient processing and good performance. However, when these methods are applied directly to remote sensing image detection, there are various problems, such as a lower detection accuracy for objects with large-scale differences and difficulty detecting small objects in complex scenes. Therefore, the network results of the YOLO series for remote sensing object detection need to be improved further to achieve better detection performance.

To address the problems of complex scenes in remote sensing images, multi-scale objects in different categories, and large scenes with small objects, we propose DFPN-YOLO, a dense feature pyramid network structure based on YOLO. Since the YOLO series became more integrated after version v3, the structure changes of network were not significant. Therefore, we use YOLOv3 as a baseline to easier compare the accuracy before and after altering the structure of network. First, we add a spatial groupwise enhancement [34] (SGE) attention module to the residual block [35] of the backbone to increase the efficiency of the backbone in extracting meaningful semantic information from complex scenes; then, we add a large detection layer to improve the accuracy in detecting small objects in remote sensing images; and finally, we propose Dense-FPN, a dense feature pyramid network structure that combines the semantic information of the feature layers to improve the ability to detect objects at different scales.

The remainder of this paper is organized as follows: related work on YOLO, in particular the framework structure of YOLOv3, is discussed in Section 2. In Section 3, our methodology is described in detail. In Section 4, an experimental validation is presented, introducing the datasets used as well as the relevant evaluation metrics. Finally, the conclusions are given in Section 5.

2. Related Work

YOLO was first proposed by Joseph Redmon et al. in 2015, and the official version has been updated from YOLOv1 to YOLOv3. It is worth noting that YOLOv4 and YOLOv5 are not official versions. The YOLO series network directly regresses the information of a grid cell bounding box to the final feature map, yielding three prediction values for each bounding box: (1) the probability of the object being in the grid; (2) the coordinates of the bounding box, and (3) the object class and its probability. For each grid cell, the predicted values include five parameters: x, y, w, h, and cf, where x, y, w, and h denote the x and y coordinates, height, and width of the center point of the enclosing box, respectively, and cf denotes the confidence of the bounding box. Therefore, the loss function of the whole network can be written as shown in Equation (1):

$$\begin{aligned} \text{loss} = & \; r_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} g_{ij}^{obj} [(x_i - \hat{x}_i^j)^2 + (y_i - \hat{y}_i^j)^2] \\ & + r_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} g_{ij}^{obj} [(\sqrt{w_i^j} - \sqrt{\hat{w}_i^j})^2 + (\sqrt{h_i^j} - \sqrt{\hat{h}_i^j})^2] \\ & - \sum_{i=0}^{s^2} \sum_{j=0}^{B} g_{ij}^{obj} [\hat{C}_i^j \log(C_i^j) + (1 - \hat{C}_i^j) \log(1 - C_i^j)] \\ & - r_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^{B} g_{ij}^{noobj} [\hat{C}_i^j \log(C_i^j) + (1 - \hat{C}_i^j) \log(1 - C_i^j)] \\ & - \sum_{i=0}^{s^2} g_{ij}^{obj} \sum_{c \in classes} [\hat{P}_i^j \log(P_i^j) + (1 - \hat{P}_i^j) \log(1 - P_i^j)] \end{aligned} \tag{1}$$

In the equation, s^2 represents the number of grids, B represents the number of anchors, and γ_{ij}^{obj} represents whether the corresponding anchor box is responsible for detecting the object. If it is responsible, γ_{ij}^{obj} is 1; otherwise, it is 0. $\hat{C}_i^j$ represents the ground truth, which is determined by whether or not the bounding box of the grid is responsible for predicting an object. If this is the case, $\hat{C}_i^j$ is 1; otherwise, it is 0. When calculating the multi-classification loss, we regard it as multiple two-classification tasks. For each category, the ground truth $\hat{P}_i^j$ is 1 if the object belongs to this category; otherwise, it is 0, and the prediction P_i^j indicates the probability that the object belongs to this category. Our approach follows the loss function of YOLOv3, which will not be described in subsequent sections.

The backbone of YOLOv3 is Darknet53 [36], which downsamples each input image five times, with the last three downsampled layers transmitted to the detection layer for object detection after feature fusion. The structure of the YOLOv3 is shown in Figure 1. For a 416 × 416 input image, the three scales of the detection layers are 13 × 13, 26 × 26, and 52 × 52, which are responsible for detecting objects at different scales. The deep layer contains a large amount of semantic information, while the shallow feature-mapping layer contains a large amount of fine-grained information. Therefore, the network uses a feature pyramid to perform feature fusion, where the downsampled 32-fold feature map is first upsampled to the same size as the downsampled 16-fold feature map, and then, the feature maps are cascaded together. Similarly, the same process is performed for the downsampled 16-fold feature map and the downsampled 8-fold feature map.

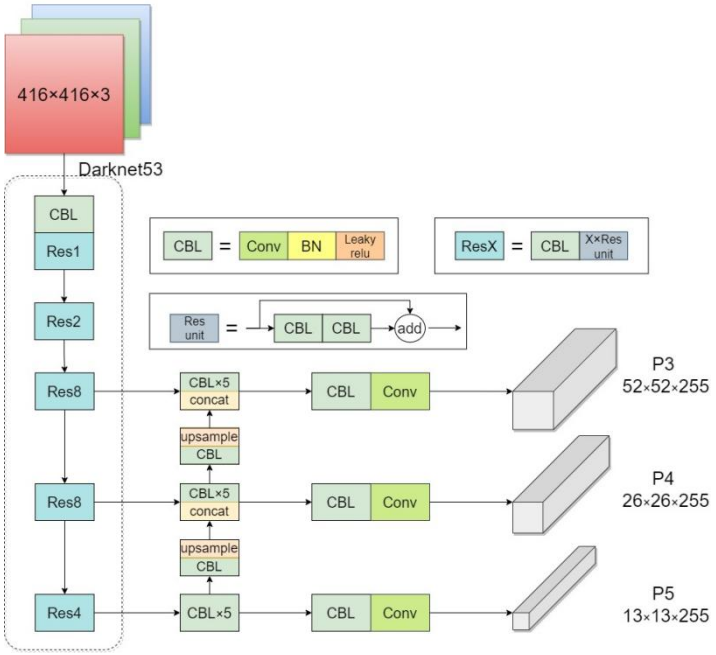


Figure 1. The structure of YOLOv3. BN in the figure represents batch normalization.

3. Methods

Even the YOLOv3 has a poor performance in remote sensing object detection. Because remote sensing images are characterized by complex scenes, small objects, and multi-scale objects in different categories, additional detection layers are necessary in remote sensing object detection to extract features more efficiently without deepening the network. For this purpose, we propose the DFPN-YOLO. The structure of DFPN-YOLO is shown in Figure 2.

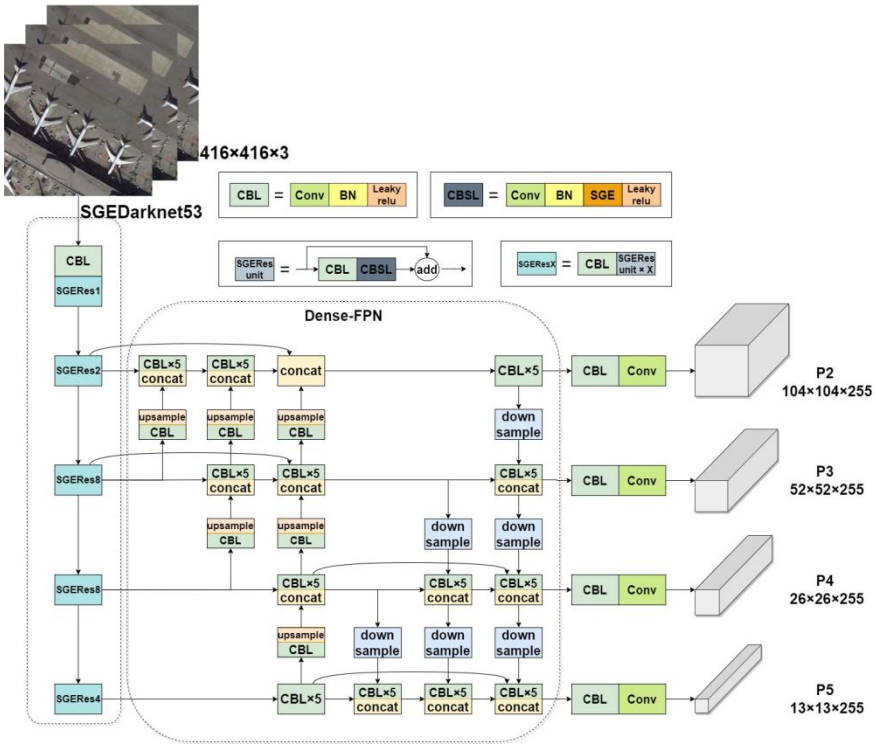


Figure 2. The structure of DFPN-YOLO. BN in the figure represents batch normalization.

The specific methods are as follows: first an attention module is added to the residual block of the backbone to allow the network to more effectively extract features in complex scenes. Second, a larger detection layer is added on top of the original three detection layers to allow the network to detect small objects. The four detection layers correspond to 4×, 8×, 16×, and 32× downsampling of the original image, and the feature information of small objects is fully retained on the feature map with 4× downsampling. Finally, a dense feature pyramid network structure is used to combine the scales of the four feature layers, allowing the fused feature layers to combine semantic information before and after sampling, improving the object detection performance at different scales.

3.1. Attention-Based Feature Extraction Network

Darknet53, the backbone of YOLOv3, is mainly composed of residual units, and because of the way these residuals are combined, Darknet53 can be trained effectively even when stacked to 53 layers, with no gradient explosions or gradient disappearance. However, because the residual block stacking is very deep, the training is slow, and the shortcut in the individual residual blocks causes the perceptual field to capture only detail information and not global characteristics. Thus, in complex scenes, the features in each layer are not extracted sufficiently or effectively, and complex scenes in remote sensing image object detection and the simple stacking of residual units to deepen the network do not significantly improve the feature extraction ability. In order to solve the problem, which is difficult to extract features under the complex background of remote sensing images, we add the spatial groupwise enhancement (SGE) attention module to the residual unit. SGE is based on SE-Net and combined with the idea of grouping so that it is a lightweight attention module that increases the classification and detection performance with nearly no increase in the number of parameters or the computational cost. A complete feature is composed of many subfeatures, which are distributed in groups in each layer; however, these subfeatures are all processed in the same manner and are all affected by background noise, which can lead to incorrect recognition and localization results. Therefore, the addition of the SGE module can generate an attention factor in each group, allowing the importance of each subfeature to be obtained and each group to learn and suppress noise as follows:

1. The feature map is divided into G groups based on the channel dimension;
2. The attention factor of each group is determined;
3. Global average pooling is performed on each group to obtain the vector g;
4. The vector g is element-wise dotted with the original group feature;
5. The vector is normalized, sigmoid activated, and element-wise dotted with the original group feature;
6. Finally, the enhanced feature map is generated.

A feature map was obtained from the original image after continuous processing of multiple convolutions, and then, it is divided into several groups along the channel dimension and processed by SGE module. The attention factor of each group of features was obtained and mapped to the corresponding feature map. Finally, after semantic feature enhancement, the feature map was generated. The SGE structure diagram is shown in Figure 3.

Due to the light weight of the SGE module and its effectiveness for higher-order semantic features, the SGE module can be perfectly integrated with Darknet53. We add the SGE module to the residual unit to improve the ability of the backbone network to extract features in complex scenes. In particular, the original feature map is convolved, batch normalized and activated by the activation function, and after the second convolution and batch normalization, the feature enhancement is performed by the SGE module, and the enhanced feature map is summed with the original feature map by shortcut edges and then activated by the activation function. Figure 4 shows the SGE module after it has been inserted into the residual block.

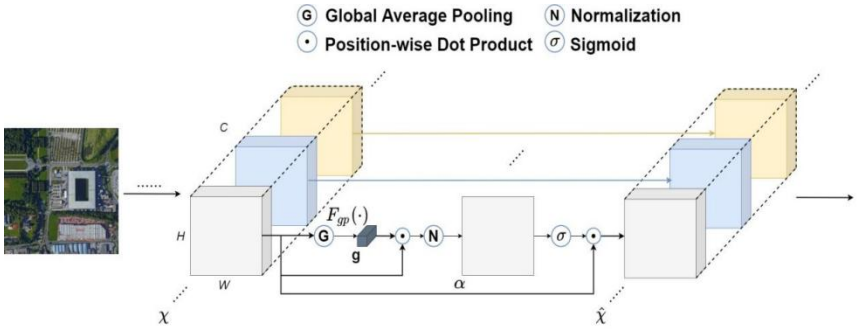


Figure 3. The structure of the SGE module.

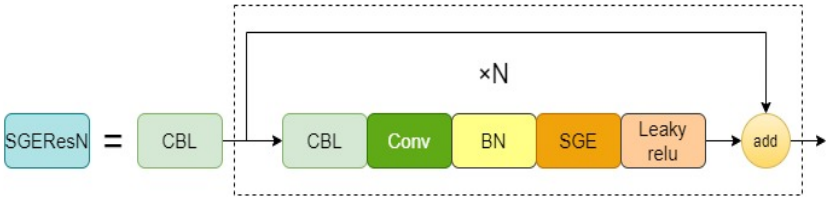


Figure 4. The structure of SGEResN. BN represents batch normalization, and CBL represents conv, BN, and leaky relu.

The residual units with attention modules are continuously stacked to form the backbone SGEDarknet53, whose structure is shown in Table 1.

Table 1. The parameters of SGEDarknet53, where k represents kernel size, s represents stride, and p represents padding.

Layers	Filters	Size	Output Size
Convolutional	32	k = 3, s = 1, p = 1	416 × 416 × 32
Convolutional	64	k = 3, s = 2, p = 1	208 × 208 × 64
Convolutional	32	k = 1, s = 1, p = 0	208 × 208 × 32
SGEresidual	64	k = 3, s = 1, p = 1	208 × 208 × 64
Convolutional	128	k = 3, s = 2, p = 1	104 × 104 × 128
2 × Convolutional	64	k = 1, s = 1, p = 0	104 × 104 × 64
2 × SGEresidual	128	k = 3, s = 1, p = 1	104 × 104 × 128
Convolutional	256	k = 3, s = 2, p = 1	52 × 52 × 256
8 × Convolutional	128	k = 1, s = 1, p = 0	52 × 52 × 128
8 × SGEresidual	256	k = 3, s = 1, p = 1	52 × 52 × 256
Convolutional	512	k = 3, s = 2, p = 1	26 × 26 × 512
8 × Convolutional	256	k = 1, s = 1, p = 0	26 × 26 × 256
8 × SGEresidual	512	k = 3, s = 1, p = 1	26 × 26 × 512
Convolutional	1024	k = 3, s = 2, p = 1	13 × 13 × 1024
4 × Convolutional	512	k = 1, s = 1, p = 0	13 × 13 × 512
4 × SGEresidual	1024	k = 3, s = 1, p = 1	13 × 13 × 1024

3.2. Detection Layer for Small Objects

YOLOv3 uses different detection layers to detect objects of various sizes. For a 416 × 416 input image, the sizes of the three detection layers are 13 × 13, 26 × 26, and 52 × 52, i.e., the feature maps of the three detection layers are downsampled 8 times, 16 times, and 32 times, respectively. The smaller the size of the feature map, the larger the area corresponding to each grid cell in the input image; in contrast, the larger the size of the feature map, the smaller the area corresponding to each grid cell in the input image.

Thus, the 13 × 13 detection layer is suitable for detecting large objects, while the 52 × 52 detection layer is suitable for detecting small objects. However, when compared with the original map, the 52 × 52 detection layer is downsampled 8 times, i.e., when the size of the object is smaller than 8 × 8, the space it occupies in the feature map may be less than 1 pixel after the feature extraction process, which makes it difficult to detect small objects. In general, remote sensing images contain a large number of small objects. To further improve the detection capability of small objects in remote sensing images, one of the most direct and effective ways is to perform object detection directly on the feature map with larger resolution. Although it will increase the computational cost to a certain extent, but in the feature fusion stage, the feature maps under high resolution have relatively low dimension, the increase in the number of parameters is only concentrated in the prediction layer so that the increase in the number of parameters is relatively limited. We add a 104 × 104 detection layer to detect small objects, and compared with the original image, it downsampled four times. Theoretically, even the resolution is 4 × 4, the feature information can also be retained on this detection layer, which greatly improves the detection performance of small objects. The improved network structure with the 104 × 104 × 255 small object detection layer is called P2 layers in Figure 2.

3.3. Multiscale Feature Fusion Based on Dense Feature Pyramids

In the feature fusion stage, YOLOv3 uses a feature pyramid network [37] (FPN) to laterally combine the semantic information of the last three feature layers sampled; the feature pyramid network structure is shown in Figure 5.

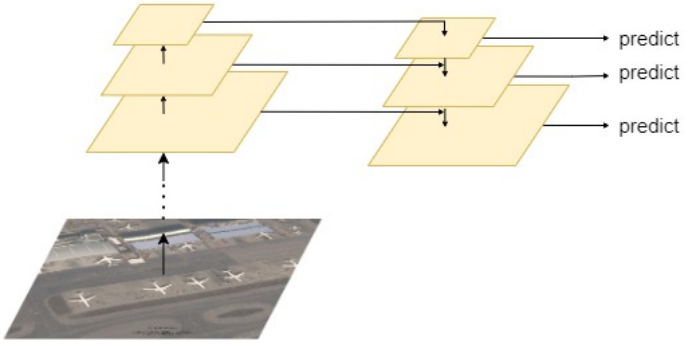


Figure 5. The structure of FPN.

However, when a P2 detection layer is added, the FPN has four layers, and the simple horizontal connection does not combine the semantic feature information well. Thus, we propose a dense feature pyramid network called Dense-FPN. Dense-FPN continuously samples and combines the feature maps of the C2, C3, C4, and C5 layers to generate the P2, P3, P4, and P5 layers. The specific approach is to upsample and combine the feature maps of the C3, C4, and C5 layers and then upsample and combine the fused feature maps with the previous layers until the top layer, C2, is reached, thus generating the middle hidden layers H2, H3, H4, and H5. After that, the feature maps of the middle hidden layers, H2, H3, and H4, are downsampled and fused with the feature maps of the next layer. The fused feature maps are downsampled and fused with the next layer until layer H5 is reached, thus generating the final layers, P2, P3, P4, and P5. We also connect the input feature layer, the hidden layer, and the output layer with a jump connection to achieve feature reuse. This connection is more conducive to gradient backpropagation, as it better utilizes the feature information and improves the information transfer efficiency between the layers. The Dense-FPN structure is shown in Figure 6.

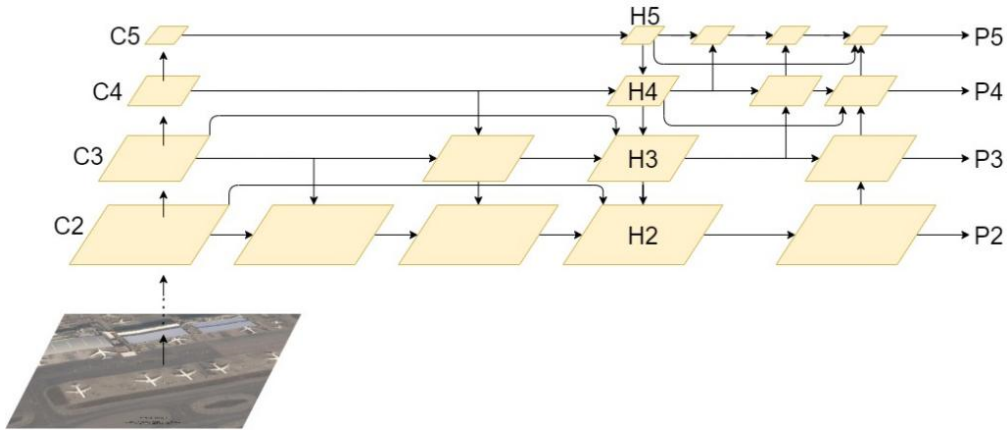


Figure 6. The structure of Dense-FPN.

3.4. K-Means for Anchor Boxes

We use the k-means algorithm to generate anchors for the four detection layers. The k-means algorithm generates anchors that have large IOUs with the ground truth, which is more conducive to network convergence. The specific method is as follows:

1. Randomly select some points as centroids of cluster for the initial aggregation, with the centroid of the cluster corresponding to the center of the sample that we will approach;
2. For each sample in the datasets, calculate the ground truth to the centroid of each cluster, and classify the sample into the cluster with the smallest distance, as shown in Equations (2) and (3), where bbox represents the bounding box, and d(bbox, centriod) represents the distance between the centroid of the cluster and the center of the bbox;

$$\mathrm{d}(\mathrm{bbox}, \mathrm{centriod}) := 1 - \mathrm{IOU}(\mathrm{bbox}, \mathrm{centriod}) \tag{2}$$

$$\mathrm{IOU} := \frac{\mathrm{S}_{\mathrm{overlap}}}{\mathrm{S}_{\mathrm{union}}} \tag{3}$$

3. Recalculate the cluster center for each cluster;
4. Repeat steps 2 and 3 until the clusters converge.

For resolution 416 × 416 input images, with the k-means algorithm, we generated 12 anchor boxes for the four detection layers: (21, 25), (25, 31), (33, 39), (44, 51), (59, 81), (84, 95), (104, 116), (119, 148), (161, 184), (221, 201), (246, 213), and (259 278). The anchor boxes (21, 25), (25, 31), and (33, 39) were designed for the added 104 × 104 detection layer, and they can be used to detect small objects, which are usually only a few pixels in size, in remote sensing images. For medium-sized objects, a slightly larger anchor can be used on a 52 × 52 or 26 × 26 feature map. The anchor boxes (221, 201), (246, 213), and (259 278) were designed for the big objects on 13 × 13 feature map. Therefore, even if an image contains objects of different sizes, as shown in Figure 7, the anchor of hierarchical designed can match the objects.

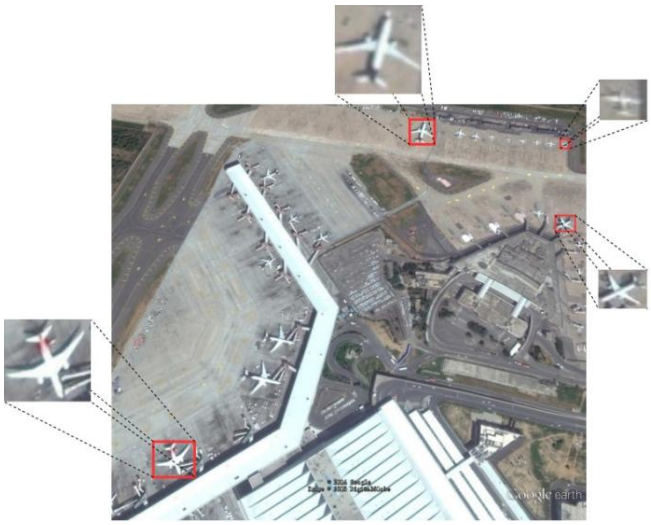


Figure 7. K-means algorithm is used to generate matching anchors for objects of different sizes.

4. Experiments and Results

To verify the effectiveness of our proposed method, we conduct comparison experiments using the publicly available RSOD [38] datasets and DIOR [39] datasets with different versions of YOLO, some classical detection algorithms, and our proposed method. In this section, we present the datasets used, the evaluation metrics, the experimental procedures, and the experimental results.

4.1. Datasets

The RSOD datasets are open object detection datasets for object detection in remote sensing images. The datasets include aircraft, fuel tanks, sports fields, and overpasses that have been annotated in the format of PASCAL VOC [40] datasets. The datasets are divided into four folders as follow:

1. 4993 aircraft in 446 images;
2. 191 playgrounds in 189 images;
3. 180 overpasses in 176 images;
4. 1586 oil tanks in 165 images.

Some example images from the RSOD datasets are shown in Figure 8.

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

Figure 8. Images in the RSOD datasets. (**a**) Playground; (**b**) overpass; (**c**) oil tank; and (**d**) aircraft.

We randomly divided the datasets into a training set, a validation set, and a test set according to a 6:2:2 ratio, i.e., 580 images for training, 197 images for validation, and 199 images for testing, as shown in Table 2.

Table 2. Training, validation, and test sets for each category of the RSOD datasets.

Class	Train	Val	Test
Aircraft	268	88	90
Oil tank	93	36	36
Overpass	106	35	35
Playground	113	38	38
Total	580	197	199

The DIOR datasets are large-scale benchmark datasets for object detection in optical remote sensing images. The datasets includes 23,463 images of different seasons and weather patterns, with 190,288 object instances, a uniform image size of 800 × 800, and resolutions ranging from 0.5 m to 30 m. The DIOR datasets officially provided helped us divide the training set, verification set, and test set according to the ratio of 2.5:2.5:5 as shown in Table 3 [39]. Note that one image may contain multiple object classes, so the column totals do not simply equal the sums of each corresponding column. The number of each category represents the object number, not the number of images, and the "Total" in last line represents the number of images in each set.

Table 3. Training, validation and test sets for each category of the DIOR datasets.

Class	Train	Val	Test
Airplane	344	338	705
Airport	326	327	657
Baseball field	551	557	1312
Basketball court	336	329	704
Bright	379	495	1302
Chimney	202	204	448
Dam	238	246	502
Expressway service area	279	281	565
Expressway toll station	285	299	634
Golf field	216	239	491
Ground track field	536	454	1322
Harbor	328	332	814
Overpass	410	510	1099
Ship	650	652	1400
Stadium	289	292	619
Storage tank	391	384	839
Tennis court	605	630	1347
Train station	244	549	501
Vehicle	1556	1558	3306
Windmill	403	404	809
Total	5862	5863	11,738

Some example images from the DIOR datasets are shown in Figure 9.

As shown in the figure, the scenes in the DIOR datasets and RSOD datasets are relatively complex, including scenes such as mountains, lakes, grasslands, farms, docks, and airports. The scales of the different object categories vary greatly, ranging from small objects such as airplanes and cars, with sizes less than 30 × 30, to playgrounds and golf courses, with sizes larger than 500 × 500. The scales of similar objects, such as ships and airplanes, also vary greatly.

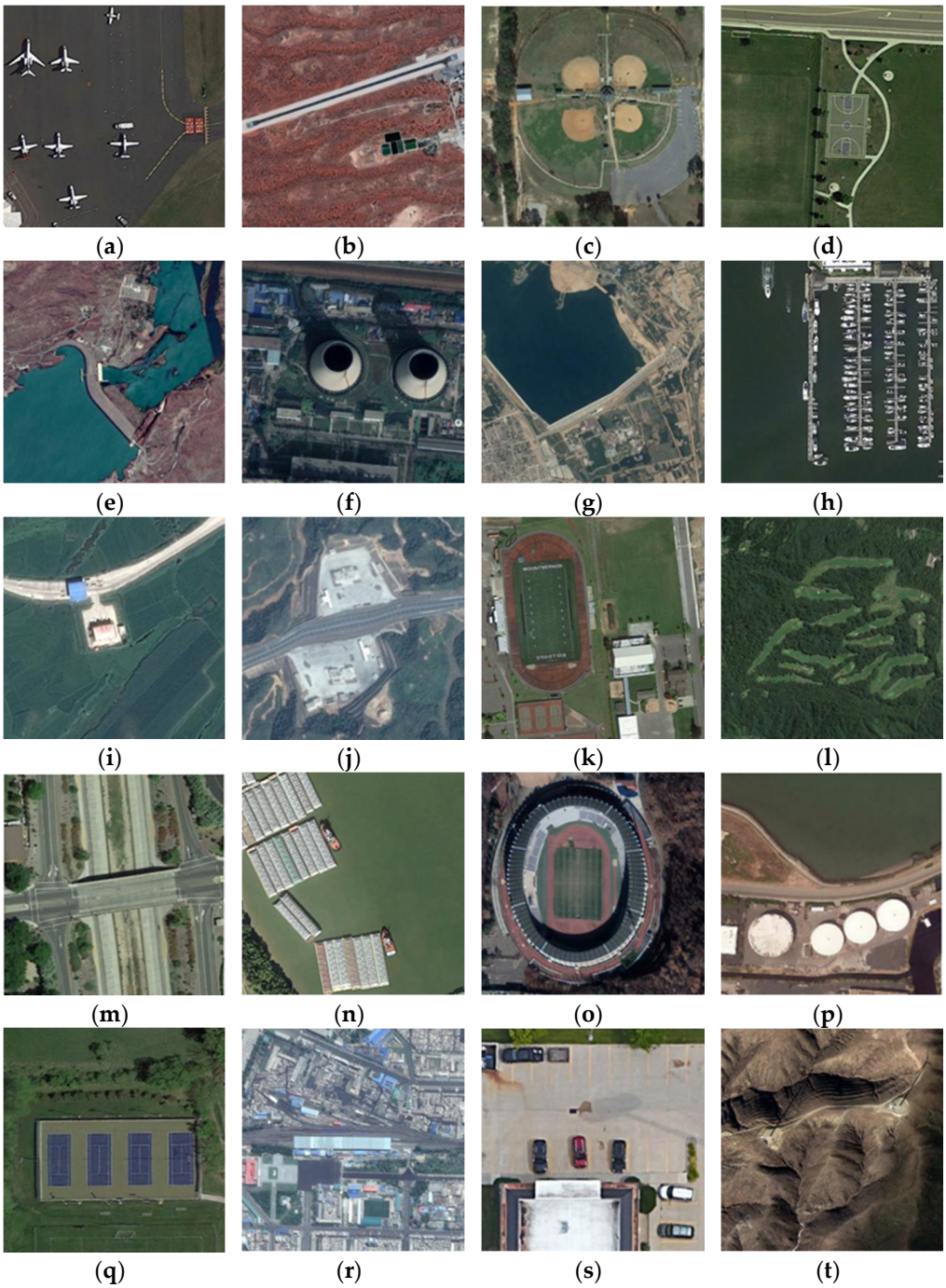


Figure 9. Images in the DIOR datasets. (**a**) airplane; (**b**) airport; (**c**) baseball field; (**d**) basketball court; (**e**) bridge; (**f**) chimney; (**g**) dam; (**h**) harbor; (**i**) expressway toll station; (**j**) expressway service area; (**k**) ground track field; (**l**) golf field; (**m**) overpass; (**n**) ship; (**o**) stadium; (**p**) storage tank; (**q**) tennis court; (**r**) train station; (**s**) vehicle; and (**t**) windmill.

4.2. Evaluation Metrics

In this paper, we use the mean average precision [41] (mAP) as an evaluation metric. The mAP is an important metric for evaluating object detection performance. We divide the samples into true-positive (TP), false-positive (FP), true-negative (TN), and false-negative (FN) cases to calculate the precision (P) and recall (R) as shown in Equation (4).

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4}$$

The precision and recall are two mutually constrained and balanced metrics. To measure these two metrics, we introduce the mAP, which is defined as the area under the average PR curve of each category at different confidence levels as shown in Equation (5).

$$\mathrm{mAP} = \frac{1}{N_C}\sum_{i=1}^{N_C}\int_0^1 P_i(R_i)dR_i \quad (5)$$

where N_C represents the number of categories in the datasets.

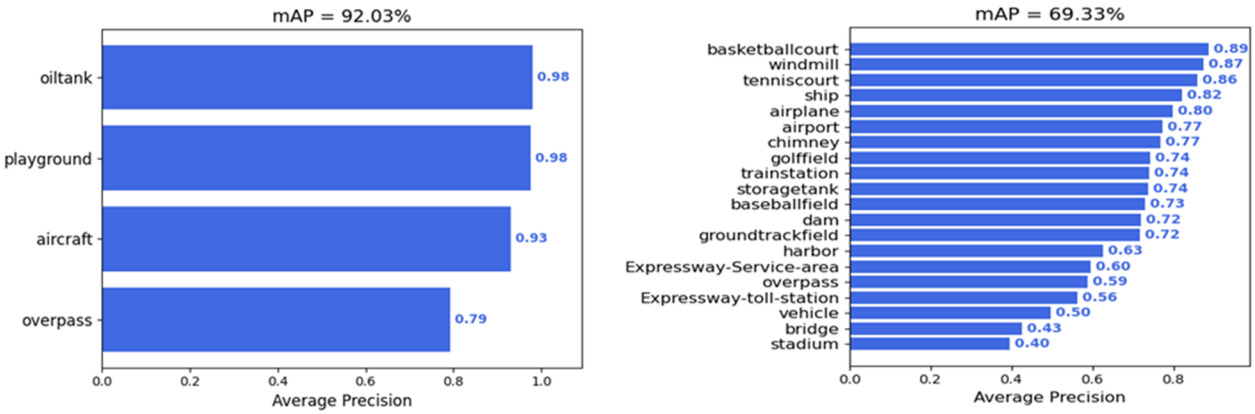
4.3. Experimental Design

We trained the RSOD datasets and DIOR datasets using Faster RCNN, SSD, YOLOv2, YOLOv3, YOLOv3-SPP, YOLOv4, and DFPN-YOLO in the PyTorch framework and performed data augmentation uniformly for the unbalanced categories of the original datasets. All experiments were performed on four NVDIA GTX 2080Ti with 11 GB of RAM, and to ensure the fairness of the comparison experiments, we used stochastic gradient descent [42] (SGD) to optimize the model with a momentum of 0.843 and a weight decay of 0.00036.

4.4. Results and Analysis

4.4.1. Experimental Results of DFPN-YOLO

DFPN-YOLO achieved a high performance when it was tested on the RSOD datasets and DIOR datasets. The result of each categories as shown in Figure 10.



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

Figure 10. Results of each categories on the RSOD datasets (**a**) and the DIOR datasets (**b**).

The above figures show the results of the DFPN-YOLO on the DIOR datasets and the RSOD datasets, including the average precision of the different categories and the mAP of the total categories. Our DFPN-YOLO model had a better detection performance on the RSOD datasets, but the slightly lower performance on overpass images was difficult to improve due to a fewer number of training samples. On the DIOR datasets, our model had 13 classes with AP values greater than 0.7. Some of the test results are shown in Figure 11.

However, we found that there are some categories with low detection performance on the DIOR datasets, such as vehicles, bridges, and stadiums. According to our analysis of the test set results, our DFPN-YOLO model had a large number of false positives for small dense objects, such as ships and vehicles, as shown in Figure 12.

The reason for the high number of false positives is that our model detects some small objects that are not labeled but do exist, such as vehicles and ships. Since we add a detection layer for small objects, our model detects some real objects with lower confidence, which have an impact in the calculation of mAP despite their lower confidence, resulting in a lower final accuracy. Figure 12b shows that although there are many small vehicles, no vehicles are marked in the labeled figure. However, our model detects some of the

small cars. Furthermore, there were a small number of training samples for objects such as stadiums, increasing the difficulty of training the model for these objects.

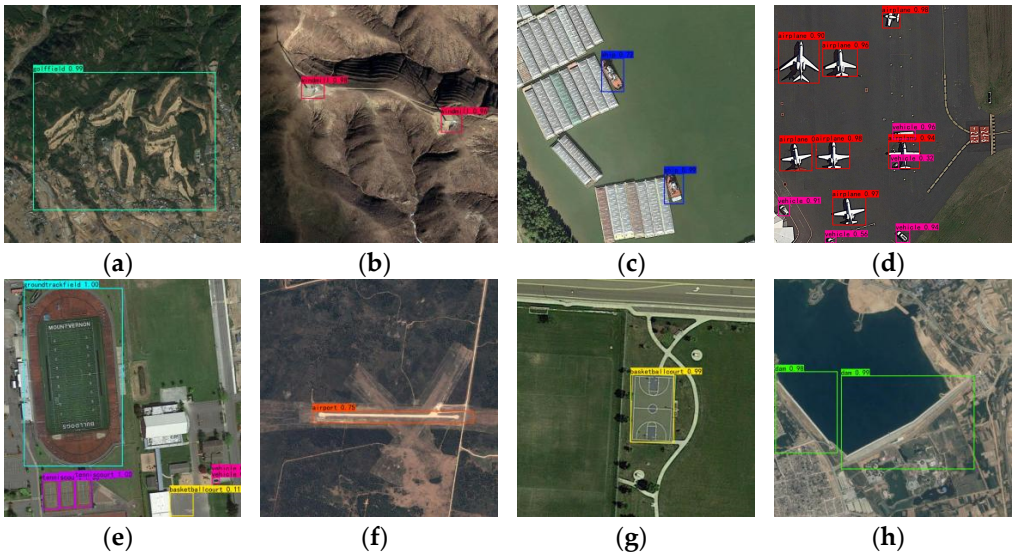


Figure 11. The detection results of DFPN-YOLO on the DIOR datasets. (**a**) Golf field; (**b**) windmill; (**c**) ship; (**d**) airplane; (**e**) ground track field; (**f**) airport; (**g**) basketball court; and (**h**) dam.

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

Figure 12. The analysis of detection results on the DIOR datasets. (**a**) Detection results of TP and FP; (**b**) detection results of vehicles without labeled.

4.4.2. Results of the Comparison Experiment

To further validate the effectiveness of our method, we conducted comparison experiments using Faster RCNN with resnet50 as backbone, SSD with vgg16 as backbone, YOLOv2 with darknet19 as backbone, YOLOv3 with darknet53 as backbone, YOLOv3-SPP with darknet53 and spatial pyramid pooling (SPP) module as backbone, YOLOv4 with CSPdarknet53 as backbone, and DFPN-YOLO with SGEdarknet53 as backbone to compare the accuracy of the algorithms based on the mAP, and the results on the DIOR datasets are shown in Table 4.

Classes 1–20 represent the following categories: airplane, airport, baseball field, basketball court, bridge, chimney, dam, expressway service area, expressway toll station, golf field, ground track field, harbor, overpass, ship, stadium, storage tank, tennis court, train station, vehicle, and windmill. Similarly, we performed comparison experiments on the RSOD datasets, with Classes 1–4 representing the oil tank, playground, aircraft, and overpass, respectively. The results are shown in Table 5.

The experimental results show that, based on the YOLOv3, when only the SGE attention module was added, the overall detection performance of the four categories improved due to the enhanced feature extraction ability of the backbone. After the fourth detection feature layer was added for small object detection, the mAP of category three, i.e., small objects in the aircraft category, significantly increased from 88.4% to 91.4%. After the Dense-FPN structure was added, the overall detection accuracy of the four object categories of objects improved, which shows that the Dense-FPN structure has a strong feature fusion capability for objects of different scales. In addition, compared with the original YOLOv3 without improvement, the mAP improved from 83.9% to 92% after adding the SGE module, the fourth detection feature layer, and the Dense-FPN structure, demonstrating the effectiveness of our method.

Table 4. Comparison of the AP of the different methods on the DIOR datasets.

Method	Faster RCNN	SSD	YOLOv2	YOLOv3	YOLOv3-SPP	YOLOv4	Ours
Class 1	54.5	60.1	58.5	76.2	76.7	79.1	80.2
Class 2	70.2	61.8	52.4	66.9	67.2	72.7	76.8
Class 3	63.6	67.5	70.6	72.0	71.4	73.2	72.7
Class 4	82.4	59.2	66.2	85.6	86.2	88.4	89.1
Class 5	43.1	34.5	37.1	34.2	39.6	40.2	43.4
Class 6	74.7	66.0	70.0	73.6	75.3	76.3	76.9
Class 7	59.1	46.2	51.4	55.2	62.4	66.5	72.3
Class 8	65.4	57.8	55.7	56.7	55.1	58.8	59.8
Class 9	62.8	54.3	55.9	55.2	53.9	56.0	56.4
Class 10	74.9	66.8	68.9	64.1	68.3	68.1	74.3
Class 11	75.3	70.1	66.2	71.4	72.8	72.4	71.6
Class 12	44.2	26.3	42.1	51.6	52.4	57.5	63.1
Class 13	52.9	47.2	50.9	54.3	56.0	57.2	58.7
Class 14	72.2	58.4	66.2	75.2	79.6	78.8	81.5
Class 15	57.1	51.7	51.3	37.4	42.9	38.8	40.1
Class 16	51.2	50.2	49.6	66.2	62.1	70.7	74.2
Class 17	79.8	64.5	67.4	84.3	85.5	85.4	85.8
Class 18	51.3	42.3	39.3	50.7	58.7	64.4	73.6
Class 19	45.0	37.2	40.2	41.5	42.0	46.6	49.7
Class 20	80.7	62.2	55.8	75.8	79.1	83.5	86.5
mAP (%)	63.02	54.22	55.79	62.41	64.37	66.73	69.33

Table 5. Comparison of the AP of the different methods on the RSOD datasets.

Method	Faster RCNN	SSD	YOLOv2	YOLOv3	YOLOv3-SPP	YOLOv4	Ours
Class 1	90.4	70.6	69.3	86.2	90.6	94.3	97.6
Class 2	89.2	81.3	84.8	88.7	87.2	92.1	97.8
Class 3	87.6	77.5	70.7	86.1	91.5	94.6	93.3
Class 4	73.2	69.2	66.1	74.6	80.3	84.0	79.3
mAP (%)	85.1	74.7	72.7	83.9	87.4	91.3	92.0

On the DIOR datasets, our method has the highest mAP from the original 62.41% of YOLOv3 to 69.33% while outperforming other advanced methods, even higher than the 66.73% of YOLOV4, and we have the best detection performance in most categories. Of the RSOD datasets, our method is also the most accurate. Compared with 83.9% of YOLOv3, DFPN-YOLO reaches 92%, which is even 0.7% higher than YOLOv4. Furthermore, in the two categories of oil tank and playground, our detection performance is much higher than other methods, with AP reaching nearly 98%.

4.4.3. Ablation Experiments

To further validate the improved performance of Dense-FPN structure, we verified the effective improvement introduced by each step of our method by performing ablation experiments on the RSOD datasets. The results are shown in Table 6.

Table 6. Ablation experiments on the RSOD datasets.

Experiment	Exp 1	Exp 2	Exp 3	Exp 4
SGE		√	√	√
Scale 4			√	√
DFPN				√
Class 1	86.2	90.2	91.7	97.6
Class 2	88.7	91.6	92.2	97.8
Class 3	86.1	88.4	91.4	93.3
Class 4	74.6	75.3	75.0	79.3
mAP (%)	83.9	86.4	87.8	92.0

5. Conclusions

As satellite imaging technology and deep learning technology have developed, remote sensing object detection has become a popular research topic. To address the problems of complex scenes, large scenes with small objects, and large-scale differences of objects in remote sensing object detection, a dense feature pyramid network based on YOLO known as DFPN-YOLO was proposed in this paper.

First, we added an attention module to the residual blocks of the backbone to allow the network to quickly extract key feature information in complex scenes. Then, we added a larger detection layer to address the difficulty of detecting small objects in large fields of view. Finally, we proposed a dense feature pyramid network structure named Dense-FPN, which enabled all four detection layers to combine the semantic information, improving the object detection performance at different scales. Our proposed method achieves a high accuracy on the RSOD datasets and DIOR datasets and outperforms both classical algorithms and even outperforms the YOLOv4 in terms of the mAP metric. On the DIOR datasets, our algorithm achieves a maximum mAP of 69.33%, which is considerably higher than the 62.41% mAP of YOLOv3, and due to the Dense-FPN structure, the detection accuracy of our algorithm is higher than the accuracies of other algorithms in most object categories. On the RSOD datasets, the precision of our algorithm is better than the performance of other classical algorithms, reaching an mAP of 92%, which is 8% higher than the mAP of 83.9% of YOLOv3. From the comparison experiments, we found that YOLOv4 with an FPN + PAN structure and DFPN-YOLO with a Dense-FPN structure significantly outperformed YOLOv3 in terms of overall performance, demonstrating the importance of feature fusion for detection precision. Furthermore, our method performed slightly better than YOLOv4.

However, although our method achieves good performance on the RSOD datasets and DIOR datasets, it has a poor detection performance on some high-noise remote sensing images, and the detection of blurred images and high-noise remote sensing images remains a major challenge for remote sensing object detection. We will carry out additional research in future work.

Author Contributions: Conceptualization, Y.S. and F.B.; methodology, Y.S. and F.B.; software, Y.S. and Y.G.; validation, F.B., W.L., and X.H.; formal analysis, X.H.; investigation, X.H.; data curation, Y.G.; writing—original draft preparation, Y.S.; writing—review and editing, F.B. and W.L.; supervision, Y.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 61971006).

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. Chen, W.; Zhou, S.; Pan, Z.; Zheng, H.; Liu, Y. Mapless Collaborative Navigation for a Multi-Robot System Based on the Deep Reinforcement Learning. *Appl. Sci.* **2019**, *9*, 4198. [CrossRef]
2. Tang, S.; Chen, Z. Understanding Natural Disaster Scenes from Mobile Images Using Deep Learning. *Appl. Sci.* **2021**, *11*, 3952. [CrossRef]
3. Zhao, Y.; Deng, X.; Lai, H. A Deep Learning-Based Method to Detect Components from Scanned Structural Drawings for Reconstructing 3D Models. *Appl. Sci.* **2020**, *10*, 2066. [CrossRef]
4. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv* **2013**, arXiv:1312.6034. [CrossRef]
5. Kaut, H.; Singh, R. A Review on Image Segmentation Techniques for Future Research Study. *Int. J. Eng. Trends Technol.* **2016**, *35*, 504–505. [CrossRef]
6. Li, R.; Zeng, B.; Liou, M.L. A new three-step search algorithm for block motion estimation. *IEEE Trans. Circuits Syst. Video Technol.* **2002**, *4*, 438–442.
7. Benfold, B.; Reid, I. Stable multi-target tracking in real-time surveillance video. In Proceedings of the Computer Vision & Pattern Recognition (CVPR 2011), Colorado Springs, CO, USA, 20–25 June 2011.
8. Cheng, G.; Han, J. A Survey on Object Detection in Optical Remote Sensing Images. *ISPRS J. Photogramm. Remote Sens.* **2016**, *117*, 11–28. [CrossRef]
9. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2999–3007.
10. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1627–1645. [CrossRef]
11. Divvala, S.K.; Efros, A.A.; Hebert, M. How important are Deformable Parts in the Deformable Parts Model? In Proceedings of the European Conference on Computer Vision, Florence, Italy, 7–13 October 2012.
12. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–26 June 2005; pp. 886–893.
13. Gunn, S.R. Support vector machines for classification and regression. *ISIS Tech. Rep.* **1998**, *14*, 5–16.
14. Ferrigno, P. Regulated nucleo/cytoplasmic exchange of HOG1 MAPK requires the importin β homologs NMD5 and XPO1. *EMBO J.* **2014**, *17*, 5606–5614. [CrossRef]
15. Roska, T.; Chua, L.O. The CNN universal machine: An analogic array computer. *IEEE Trans. Circuits Syst. II Analog. Digit. Signal Process.* **2015**, *40*, 163–173. [CrossRef]
16. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767. [CrossRef]
17. Cui, Z.; Li, Q.; Cao, Z.; Liu, N. Dense Attention Pyramid Networks for Multi-Scale Ship Detection in SAR Images. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 8983–8997. [CrossRef]
18. Huang, W.; Li, G.; Chen, Q.; Ju, M.; Qu, J. CF2PN: A Cross-Scale Feature Fusion Pyramid Network Based Remote Sensing Target Detection. *Remote Sens.* **2021**, *13*, 847. [CrossRef]
19. Xu, D.; Wu, Y. FE-YOLO: A Feature Enhancement Network for Remote Sensing Target Detection. *Remote Sens.* **2021**, *13*, 1311. [CrossRef]
20. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014.
21. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
22. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
23. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Fu, C.; Berg, A.C. SSD: Single Shot Multibox Detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
24. RScott. FCLIP demos improved SSDS detect-to-engage co-ordination. *Jane's Int. Def. Rev.* **2016**, *49*, 17.
25. Bai, G.; Hou, J.; Zhang, Y.; Li, B.; Han, H.; Wang, T.; Hinkelmann, R.; Zhang, D.; Guo, L. An intelligent water level monitoring method based on SSD algorithm. *Measurement* **2021**, *185*, 110047. [CrossRef]
26. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
27. Shaifee, M.J.; Chywl, B.; Li, F.; Wong, A. Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video. *arXiv* **2017**, arXiv:1709.05943. [CrossRef]

28. Chen, Q.; Wang, Y.; Yang, T.; Zhang, X.; Cheng, J.; Sun, J. You only look one-level feature. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3039–13048.
29. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR 2019, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
30. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
31. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
32. Bochkovskiy, A.; Wang, C.Y.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934v1.
33. 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 (ICCV) Workshops, Montreal, BC, Canada, 11–17 October 2021.
34. Li, X.; Hu, X.; Yang, J. Spatial group-wise enhance: Improving semantic feature learning in convolutional networks. *arXiv* **2019**, arXiv:1905.09646.
35. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
36. Wang, H.; Zhang, F.; Wang, L. Fruit classification model based on improved Darknet53 convolutional neural network. In Proceedings of the 2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Vientiane, Laos, 11–12 January 2020; pp. 881–884.
37. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Venice, Italy, 22–29 October 2017; pp. 2117–2125.
38. Xiao, Z.; Liu, Q.; Tang, G.; Zhai, X. Elliptic Fourier transformation-based histograms of oriented gradients for rotationally invariant object detection in remote-sensing images. *Int. J. Remote Sens.* **2015**, *36*, 618–644. [CrossRef]
39. Li, K.; Wan, G.; Cheng, G.; Meng, L.; Han, J. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS J. Photogramm. Remote Sens.* **2020**, *15*, 296–307. [CrossRef]
40. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
41. Cheng, G.; Zhou, P.; Han, J. Learning Rotation-Invariant Convolutional Neural Networks for Target detection in VHR Optical Remote Sensing Images. *IEEE Geosci. Remote Sens.* **2016**, *54*, 7405–7415. [CrossRef]
42. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.