remote sensing

MDPI

Article

Priority Branches for Ship Detection in Optical Remote Sensing Images

Yijia Zhang, Weiguang Sheng, Jianfei Jiang, Naifeng Jing, Qin Wang * and Zhigang Mao

Department of Micro/Nano Electronics, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; zhangyijia@sjtu.edu.cn (Y.Z.); wgshenghit@sjtu.edu.cn (W.S.); jiangjianfei@sjtu.edu.cn (J.J.); sjtuj@sjtu.edu.cn (N.J.); maozhigang@sjtu.edu.cn (Z.M.)

* Correspondence: qinqinwang@sjtu.edu.cn; Tel.: +86-21-34204546-1039

Received: 13 March 2020; Accepted: 4 April 2020; Published: 8 April 2020

Abstract: Much attention is being paid to using high-performance convolutional neural networks (CNNs) in the area of ship detection in optical remoting sensing (ORS) images. However, the problem of false negatives (FNs) caused by side-by-side ships cannot be solved, and the number of false positives (FPs) remains high. This paper uses a DLA-34 network with deformable convolution layers as the backbone. The network has two priority branches: a recall-priority branch for reducing the number of FNs, and a precision-priority branch for reducing the number of FPs. In our single-shot detection method, the recall-priority branch is based on an anchor-free module without non-maximum suppression (NMS), while the precision-priority branch utilizes an anchor-based module with NMS. We perform recall-priority branch functions based on the output part of the CenterNet object detector to precisely predict center points of bounding boxes. The Bidirectional Feature Pyramid Network (BiFPN), combined with the inference part of YOLOv3, is used to improve the precision of precision-priority branch. Finally, the boxes from two branches merge, and we propose priority-based selection (PBS) for choosing the accurate ones. Results show that our proposed method sharply improves the recall rate of side-by-side ships and significantly reduces the number of false alarms. Our method also achieves the best trade-off on our improved version of HRSC2016 dataset, with 95.57% AP at 56 frames per second on an Nvidia RTX-2080 Ti GPU. Compared with the HRSC2016 dataset, not only are our annotations more accurate, but our dataset also contains more images and samples. Our evaluation metrics also included tests on small ships and incomplete forms of ships.

Keywords: ship detection; optical remote sensing images; priority branch; side-by-side ships

1. Introduction

Advanced aerospace technologies and optical remote sensing image sensors make it possible to record images of higher resolution over larger areas. Due to the high quality of these remote sensing images, people can complete many tasks and applications that were not possible in the past. For example, Li et al. [1] used remote sensing images to perform urban flood mapping. Sun et al. [2] analyzed the number and locations of cotton bolls based on 3-D photogrammetric mapping. As the importance of ship detection increases in both military and civil use, much research is taking place in this field. In military action and rescue work, detection results can be used to find whether ships exist in a certain area and possibly their exact location. Enhancing the detection accuracy for locating ships is a hot topic worldwide.

Despite the great achievements scientists have made in enabling satellites to provide high-resolution images, partial joggles of images can reduce the accuracy of detection results. Environmental factors, such as ocean waves and severe weather, can also cause the images of ships to suffer from different levels of incompleteness. In addition, the type, color, and material of a ship [3] can also hinder the detection results. Moreover, multi-scale ships and the existence of ship-like floating objects on the sea also negatively affect detection technology. In short, methods to solve the problems involved in ship detection are urgently needed.

The earliest work on ship detection can be traced back to the 1980s when Barnum [4] presented an overview of ship detection by high-frequency skywave backscatter over-the-horizon radar. After that, works on ship detection began to spring up. There are three types of ship detection methods: traditional, machine learning-based, and deep learning-based methods.

In traditional detection, researchers applied mathematical analyses to pixel distribution, statistic data and local gradience. Although high-resolution remote sensing (RS) images can be produced by advanced sensors, such detection is not robust enough for images of different brightness and contrast. Therefore, in traditional method there is pre-processing, including land mask, image enhancement, image segmentation, and false alarm elimination [5]. After processing, Nocak et al. [6] used a global thresholding algorithm to complete ship detection. In the work of Vachon et al. [7], K distribution was used as a marine clutter distribution model. To build a fractal model of ships, Kaplan [8] introduced Hurst parameters and proposed extended fractal features for identifying targets of specific sizes in RS images. Aware of the important role of trails in detecting ships, Copeland et al. [9] described a feature space linear detection method using the local Radon transform to detect trails. However, such methods have weak anti-noise capability, and the detection result depends on the acquisition of prior knowledge.

Detection methods based on machine learning perform better when they rely on noise. One of the most popular algorithms is support vector machine (SVM). According to Dong et al. [3], a trainable Gaussian SVM classifier was performed to validate real ships from ship candidates. Inspired by Bayesian decision theory, Proia et al. [10] fixed the size of the analysis window and the threshold used to make a choice. In addition, histograms of oriented gradient (HOG) feature [11] and binary linear programming [12] were proposed to solve the problem. The machine learning-based methods achieved promising performance and outstanding accuracy [13], but the training process was complicated.

With the recent rapid development of deep learning, researchers are trying to improve the accuracy of detection with the help of convolutional neural networks (CNNs). Lin et al. [14] avoided carefully tuned parameters and complicated procedures by utilizing a full convolution network, which shows ship detection can benefit from CNN. Xie et al. [15] used the reflectance gradient to extract ship candidates and then used LFNet to verify true ships. In this work, ships under complex environment were detected well, but it took extra time to separate ship candidates from the background. Different from [15], Chen et al. [16] proposed an end-to-end structure. It enhanced the segmentation speed compared with their structure without end-to-end connection. In [17], Clément et al. benefited from their large dataset composed of 18,894 raw Synthetic Aperture Radar (SAR) images and achieved good performance in ship detection, classification, and length estimation. Considering the training dataset can be much smaller for other researchers, Dechesne's method may not work well on other datasets. To get a network which can be effective with only a small labeled dataset, Rostami et al. [18] used an innovative approach to train a DNN for classifying SAR images by transferring knowledge from an electro-optical (EO) domain. Although the method is effective, full overlap was needed between the existing classes across SAR and EO domains. Inspired by the Faster-RCNN detection framework [19], Zhang et al. [20] proved their improved Faster-RCNN could achieve a higher recall and accuracy for small ships and gathering ships. However, ships abreast with each other still cannot be detected easily. To solve the problem, Zhang et al. [21] replaced horizon boxes with inclined ship region proposals. It reduced the rate of misses to an extent, but the calculation of intersection over union (IoU) became more complicated and the method needed more skills to train. Other issues include deep learning's large amount of training time and the fact that it suffers from overfitting.

For this paper, we conducted several detecting experiments on ORS ship images and discovered the main obstacles that hindered the detection results in previous methods. These obstacles are listed below, and examples of them are shown in Figure 1.

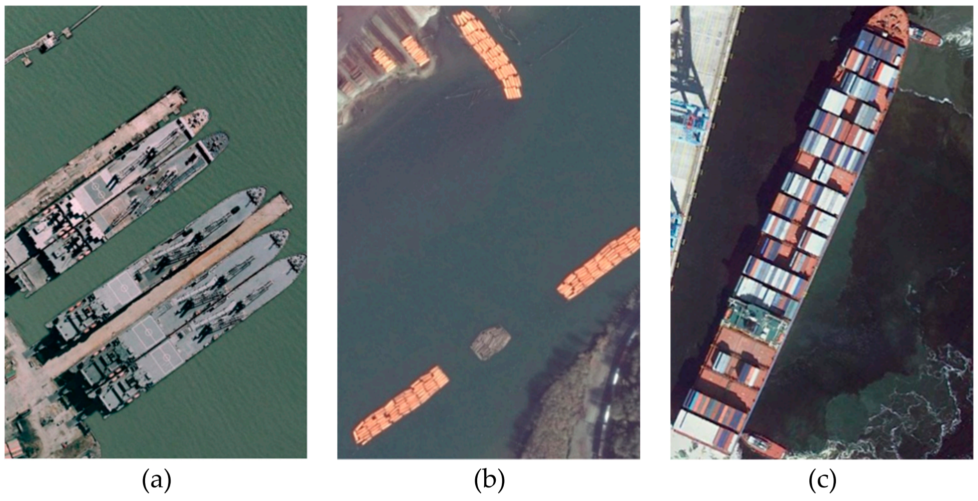

Figure 1. Main obstacles in ship detection: (**a**) side-by-side ships; (**b**) ship-like objects; (**c**) multi-scale ships.

- Side-by-side ships. In previous works, bounding boxes are produced to cover the area of ships. To avoid repeated bounding boxes on the same ship, researchers apply a non-maximum suppression (NMS) or soft-NMS strategy [22] that prevents bounding boxes from having a large overlap with the most likely box. However, the drawback to this strategy is that if ships are close together, boxes belonging to different ships will be eliminated until down to only one. Therefore, some side-by-side ships will be missed.
- Ship-like objects. Some ship-like objects will be mistaken for ships. This can be attributed to algorithms' insufficient ability to perform feature extraction, and a lack of negative samples.
- Multi-scale ships. If ships of different sizes gather, the smaller ships are often missed by detectors. Multi-scale outputs are designed to predict ships with different sizes in different modules. This method decreases the number of missed multi-scale ships but does not completely eliminate the possibility of missing them.

We use the CNN algorithm to design a single-shot detector that predicts all bounding boxes and the category at the same time. In our CNN detection method, DLA-34 [23] with deformable convolution layers is used as our backbone. During the detection process, an ORS image is first put into the backbone to create feature maps that contain all the features that the backbone extracts. Then, the feature maps are put into two branches to produce bounding boxes. One branch is the recall-priority branch, which reduces the number of missed side-by-side ships and improves the recall; the other branch distinguishes ship-like objects to increase the precision. In the end, all boxes from the two branches are filtered by priority-based selection (PBS) to obtain boxes with both high recall and high precision. The recall of multi-scale ships is also improved in this process.

Our contributions in this paper include the following:

- A state-of-the-art performance detector is proposed in this paper. Priority branches for CNN ship detectors are specially designed, and PBS is used to filter potential outputs from branches.
- To obtain more samples, we add 360 ORS ship images collected from Google Earth to the HRSC2016 dataset [24]. In our dataset, ships include warcrafts, aircraft carriers, and cargo and passenger ships. We re-annotate the dataset with consistent standards. If a ship is completely displayed in an image, we distinguish whether it is a large ship or a small ship based on whether its bounding box area is larger than 96 × 96 pixels. For those displayed incompletely, we labeled them as incomplete ships. Detecting results of large ships, small ships and incomplete ships are involved in our stricter evaluation metrics.

The code and dataset in this work will be updated to GitHub (https://github.com/Chocolife-96/Priority-Branches-for-Ship-Detection-in-Optical-Remote-Sensing-Images) in the near future.

The rest of this paper is organized as follows. Section 2 reviews the related works on ship detection and introduces our network with priority branches in detail. Experimental results are provided in Section 3. In Section 4, we give our conclusions.

2. Methodology

The architecture of our proposed priority-based ship detection is shown in Figure 2. It is based on CNN and consists of four parts: a feature fusion backbone; the recall-priority branch, which is an anchor-free module; the precision-priority branch, which is an anchor-based module; and a PBS module. After the ORS image is resized to a resolution of 512 × 512, it goes through the backbone, which integrates the multi-scale feature maps together. Then, the branch functions are performed on the same feature maps produced by the backbone. Finally, boxes from branches are filtered by the PBS to obtain predicted ones.

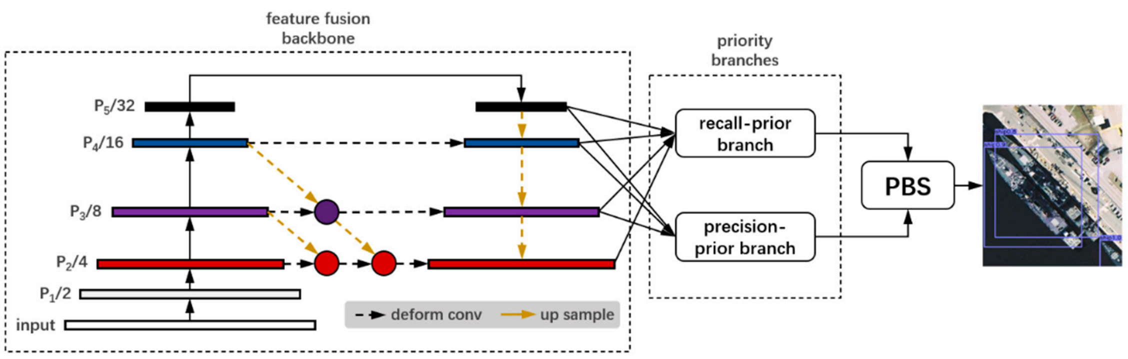

Figure 2. Our proposed priority-based architecture: DLA-34 using deformable convolution layers is selected as the backbone. Priority branches are used for improving recall rate and precision rate respectively. PBS serves as a bounding box filter which outputs only the accurate boxes we want.

2.1. Feature Fusion Backbone

Our proposed backbone is derived from deep layer aggregation (DLA) and Deformable Convolutional Networks (ConvNets). As an image classification network with hierarchical skip connection [25], DLA has the ability to integrate multi-scale feature maps together. Moreover, deformable convolution derived from Deformable ConvNets v2 is used in up-sample stages, making the convolution process sufficiently robust to work in complex environments.

2.1.1. Deep Layer Aggregation

Aggregation is the combination of different layers throughout a network. Deep layer aggregation (DLA) is when a group of aggregations are compositional and nonlinear, and the earliest layers pass through multiple aggregations. The structure of DLA can be viewed as the connections of blocks which contain several layers. At the same time, blocks are grouped into stages by feature resolution. We select the fully convolutional up-sampling DLA for dense prediction as our backbone, which is a combination of iterative deep aggregation (IDA) and hierarchical deep aggregation (HDA) [23].

The deeper the CNN layers are located, the more semantic they are. However, problems exist. Details would be lost from shallow layers, and as a result the feature maps will be coarse. Unlike the skip connections in previous work, which only integrate the shallowest layers with the deeper layers, IDA aggregates shallower layers with deeper layers at an early stage, then propagates the integrated feature maps deeper to achieve more aggregation. In this way, features from multiple levels can be integrated thoroughly.

In Equation (1) [23], *I* stands for the IDA function. Layers $x_1, \ldots, x_n$ are sorted according to their depth in a network, and N is the aggregation node

$$I(x_1, \ldots, x_n) = \begin{cases} x_1 & if\ n = 1 \\ I(N(x_1, x_2), \ldots, x_n) & otherwise \end{cases} \tag{1}$$

$$N(x_1, \ldots, x_n) = \begin{cases} \sigma(BatchNorm(\sum_i W_i x_i + b) + x_n & if\ residual\ connections\ are\ added \\ \sigma(BatchNorm(\sum_i W_i x_i + b)) & otherwise \end{cases} \tag{2}$$

Although IDA can propagate shallow features sequentially to deep layers, it is impossible to integrate features from all scales. To solve that problem, HDA is created: The tree structure is used to feed back the features from an aggregation node to the next level. This guarantees that features are integrated sufficiently.

The tree structure of HDA is present in Equation (3) [23], where N stands for aggregation node. The functions L and R are defined in Equations (4) and (5), where B is a convolution block.

$$H_n(x) = N(R^n_{n-1}(x), R^n_{n-2}(x), \ldots, R^n_1(x), L^n_1(x), L^n_2(x)) \tag{3}$$

$$L^n_2(x) = B(L^n_1(x)),\ L^n_1(x) = B(R^n_1(x)) \tag{4}$$

$$R^n_m = \begin{cases} T_m(x) & if\ m = n-1 \\ T_m(R^n_{m+1}(x)) & otherwise \end{cases} \tag{5}$$

To achieve a better fusion of local and global information, we chose the fully convolution DLA version, which makes use of interpolation with IDA. It represents a conversion from classic DLA architecture with HDA and IDA. Because of IDA, outputs from all stages are propagated to the outputs at the next stage, which promises aggregations of features from different stages. As can be seen in Figure 3, there are six stages in DLA; however, only stages 3–6 are used to get fully convoluted in this work. The structure of stage connections is depicted in Figure 4.

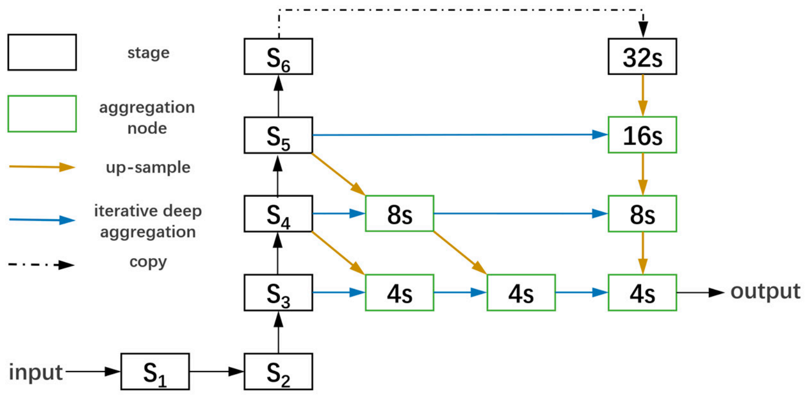

Figure 3. The full convolution DLA-34 we use. Stages 3–6 are used to go through a further level of iterative deep aggregation. IDA for interpolation is used to project and up-sample nodes by 3×3 convolution while other nodes use 1×1 convolution.

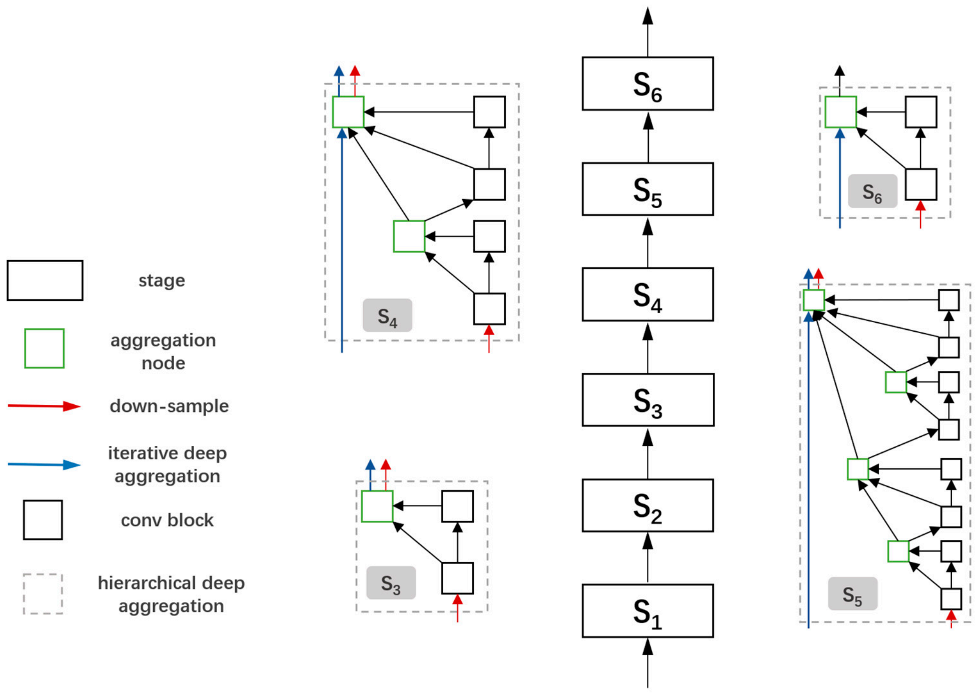

Figure 4. IDA and HDA work together to fully extract features and fuse them well. A 7 × 7 convolution and a basic block form stage 1. Stage 2 is formed by a basic block. Other stages are combinations of bottleneck blocks and aggregation nodes [23].

2.1.2. Deformable Convolution

Deformable convolution in ConvNets v2 [26] is an optimized version of deformable convolution v1 [27]. It was designed to solve the biggest challenge in object recognition and detection, caused by geometric variations due to scale, pose, viewpoint, and part deformation. In DCNv1, augmented offsets are used to select deformed sampling locations so that the feature scale in convolution layers can be expanded. Nevertheless, the vision scale that DCNv1 produces is not precise enough to cover the area of interest, which is why DCNv2 was created. A learned offset and a learned feature amplitude ensure that DCNv2 has enhanced modeling power.

In this work, we replace convolution layers in up-sampling stages of DLA-34 with DCNv2 layers. They are also used in the recall-priority branch as skip connections to further integrate features.

2.2. Priority Branches

Our experiment results presented in Section 3.5 shows that the low recall rate is mainly caused by missed side-by-side ships and multi-scale ships, while the reduced precision rate is caused by ship-like objects. Although feature fusion in the backbone can improve these phenomena to some extent, the fundamental problems are not resolved. Therefore, we thought of splitting the ship detection task into two branches. Branches are designed to improve recall and precision separately. We use an anchor-free module to work as our recall-priority branch and an anchor-based module to act as the precision-priority branch.

2.2.1. Recall-Priority Branch

Experiment results presented in Section 3.5 show that the low recall rate of side-by-side ships can be attributed to the poor precision of produced center points. Frankly speaking, NMS, as used in previous methods, is just a remedy for the imprecise center points that detections predict. Moreover, the threshold of NMS is difficult to select; while a low threshold cannot effectively curb repeated boxes of the same object, close boxes that belong to different objects may be eliminated to get down to only one bounding box. Furthermore, this problem cannot be eradicated by other improved NMS methods, such as Soft-NMS.

If the center points produced by detections are precise enough, the detecting results will not need to be filtered by NMS. Because CenterNet provides a highly effective way to obtain precise center points, it can guarantee that objects closely abreast can be recalled without using NMS. We next use the feature maps from DLA-34 to produce a center-point map $\hat{Y} \in (0,1)^{\frac{W}{R}\times\frac{H}{R}\times C}$ [26] instead of producing the coordinates of center points. C is the number of object types and R is the stride of output. In this paper, we set C to 1. According to Cao et al. [28] and Newell et al. [29], we set the default stride to R = 4, which means it will down-sample the output feature by a factor R. We resize the input image to 512 × 512, and after being processed by the branch, we obtain a heat map with the size of 128 × 128. At each location of output, five predicted parameters are produced: the key point $\hat{Y}$, offset $\hat{O}(\delta\hat{x}, \delta\hat{y})$, and size $\hat{S}$ width ($\hat{w}$) and height ($\hat{h}$) of predicted bounding box [25]. $\hat{Y}_{x,y,c} = 1$ means it is a center point of a detected ship, while $\hat{Y}_{x,y,c} = 0$ corresponds to background. At the inference time, when such a feature map is remapped to the original image, it will create errors in accuracy. Therefore, an additional local offset $\delta\hat{x}, \delta\hat{y} \in \Re^{\frac{W}{R}\times\frac{H}{R}\times 2}$ is needed for each center point.

After obtaining the heat map, we extract the peaks of detected ships by finding the top 100 points whose value is greater than (or equal to) the eight neighboring points around it. This task can be performed by applying 3 × 3 max pooling to the heat map. If the center point in a 3 × 3 matrix equals to the 3 × 3 max pooling result of the matrix, it meets the requirement of being a peak point. The $\hat{Y}_{x,y,c}$ of the top 100 points is used to indicate the confidence in potential detecting ships. Meanwhile, the location of bounding boxes is produced by Equation (6) [25], where $\hat{x}$ and $\hat{y}$ are remapped coordinates from the low resolution of heat map to the original resolution.

$$(\hat{x} + \delta\hat{x} - \frac{\hat{w}}{2}, \hat{y} + \delta\hat{y} - \frac{\hat{h}}{2}, \hat{x} + \delta\hat{x} + \frac{\hat{w}}{2}, \hat{y} + \delta\hat{y} + \frac{\hat{h}}{2}) \tag{6}$$

During training, as proposed by Law et al. [30], we map each true ground center point $p \in \Re^2 (\Re = 512)$ to the resolution of the heat map by obtaining equivalent $\widetilde{p} = \lfloor \frac{p}{R} \rfloor$. Then, we use $Y_{xyc} \in (0,1)^{\frac{W}{R}\times\frac{H}{R}\times C}$ to label the down-sampled heat map by using a Gaussian kernel, presented in Equation (7) [30]. σ_p is a standard deviation related to the size of targets.

$$Y_{xyc} = \exp\,(-\frac{(x-\widetilde{p_x})^2 + (y-\widetilde{p_y})^2}{2\sigma_p^2}) \tag{7}$$

The loss function of this branch can be divided into three parts [25]: L_h: focal loss [31] of heat map; L_{off}: L_1 loss of offset; and L_{size}: L_1 loss of object size. It is defined as

$$L_{recall} = L_h + \lambda_{off} L_{off} + \lambda_{size} L_{size} \tag{8}$$

$$L_h = -\frac{1}{N} \sum_{xyc} \begin{cases} \log\,(\,\hat{Y}_{xyc})(1-\hat{Y}_{xyc})^{\alpha} & if\ \hat{Y}_{xyc} = 1 \\ \log\,(1-\hat{Y}_{xyc})(1-Y_{xyc})^{\beta}(\hat{Y}_{xyc})^{\alpha} & otherwise \end{cases} \tag{9}$$

$$L_{off} = \frac{1}{N} \sum_{p} |\hat{O} - (\frac{p}{R} - \widetilde{p})| \tag{10}$$

$$L_{size} = \frac{1}{N} \sum_{N} |\hat{S} - S| \tag{11}$$

where λ_{off} and λ_{size} are scaling factors to confidence loss, offset loss, and size loss. α and β are hyperparameters [31] of confidence loss. We use N as the number of ships in an image. In this paper, $\lambda_{off} = 1$, $\lambda_{size} = 0.1$, $\alpha = 2$, and $\beta = 4$ [25].

2.2.2. Precision-Priority Branch

The detection results of the recall-priority branch in Section 3.5 show that the recall of side-by-side ships increases sharply, but the number of false alarms is high. This could be ascribed to mistaking ship-like floating objects on the water as ships. Meanwhile, despite the increased recall result, some ships of different sizes are still not detected. Therefore, the precision-priority branch has to make up for the false alarms and the missed ships recall-priority branch causes. Motivated by the output part of YOLOv3 [32], which draws on feature pyramid networks (FPNs) [33] and uses multiple scales to detect targets of different sizes, we use the anchor-based module to detect ships with multi-scale anchors, which can make up for the small ships missed by the recall-priority branch. To reduce the number of false alarms caused by another branch, a more powerful fusion architecture, BiFPN [34], is used to replace the feature pyramid in the YOLO module. We also use the K-means clustering method to obtain anchors based on the training boxes.

The main idea of the YOLO prediction module is to get three modules producing feature maps with different sizes after feature fusion. Using the output strides (32,16, 8) in YOLOv3 [32], we choose 16×16, 32×32, and 64×64 the sizes of the feature maps. For each feature map, three anchor priors are chosen. In order to select anchors based on the ship dataset, we apply the K-means algorithm to the annotated bounding boxes in the training dataset to obtain suitable priors [35]. In our dataset, the nine clusters are: (10, 14), (26, 22), (15, 49), (65, 30), (35, 59), (23, 149), (109, 74), (61, 151), and (143, 177). The first three clusters belong to module 1 (64×64), while the second and third three are the size of anchors belonging to modules 2 and 3, respectively.

For each location in an output feature map, the module produces three groups of data. Each group contains four coordinates $\hat{Co} = (t_x,\ t_y,\ t_w,\ t_h)$ for each bounding box and its confidence value [32]. Therefore, the dimension of each module output is S $\times$ S $\times$ N, where S = 16, 32, 64; N = B $\times$ (5+C). In this paper, B, N= 3, 1.

As shown in Figure 5, t_x and t_y can be transformed to the coordinates of the center point relative to the upper left corner of the grid cells. t_w and t_h are parameters related to the width and height p_w, p_h of the bounding box priors. The predictions correspond to

$$b_x = \sigma(t_x) + c_x,\ b_y = \sigma(t_y) + c_y \tag{12}$$

$$b_w = p_w e^{t_w},\ b_h = p_h e^{t_h} \tag{13}$$

Following the method of YOLOv3, the loss of this branch is the sum of three parts [28]: loss of class (L_{cls}), loss of location (L_{loc}), and loss of confidence (L_{conf}). L_{loc} is based on sum of error loss, while L_{cls} and L_{conf} are realized with binary cross-entropy. The overall function is

$$L_{precision} = L_{loc} + L_{cls} + L_{conf} \tag{14}$$

$$L_{loc} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{obj} |V - \hat{V}| \tag{15}$$

$$L_{cls} = -\sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{obj} (\hat{P}_i^j \ln(P_i^j) + (1 - \hat{P}_i^j) \ln(1 - P_i^j)) \tag{16}$$

$$L_{conf} = -\sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{obj} (\hat{C}_i^j \ln(C_i^j) + (1 - \hat{C}_i^j) \ln(1 - C_i^j)) - \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{noobj} (\hat{C}_i^j \ln(C_i^j) + (1 - \hat{C}_i^j) \ln(1 - C_i^j)) \tag{17}$$

where V in formulation (15) is a vector $(x_i^j,\ y_i^j,\ \sqrt{w_i^j},\ \sqrt{h_i^j})$, C, P are confidence of bounding box and classification probability. The value of I^{obj} corresponds to whether the anchor box is responsible for the detected ship, while I^{noobj} means the anchor box is out of business. λ_{coord} and λ_{noobj} are scaling factors to weigh the loss. In this experiment, $\lambda_{coord} = 1$, and $\lambda_{noobj} = 0.5$ [32].

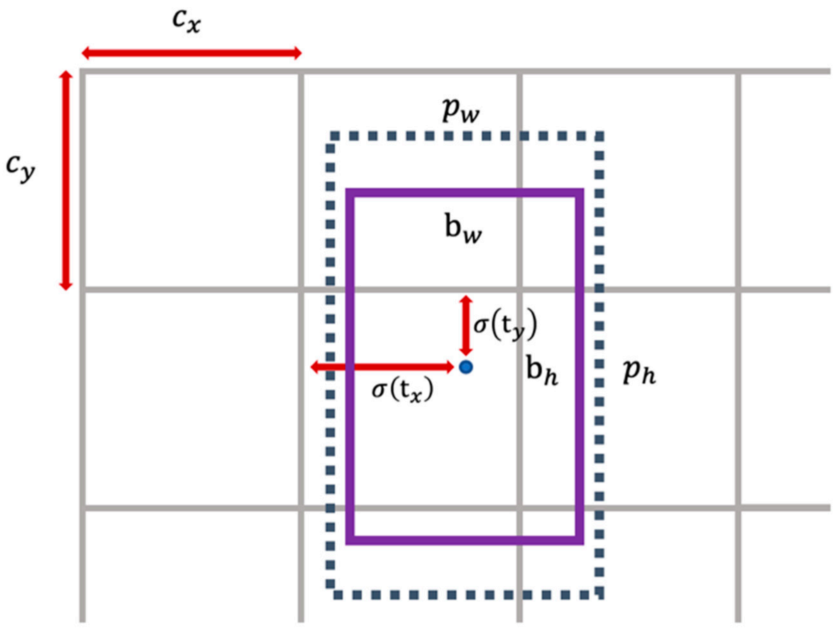

Figure 5. To be the center location of predicted bounding box, t_x and t_y should go through sigmoid function σ, which will map the value of them to the range of 0 to 1. (c_x, c_y) stands for the offset to the upper left corner of the grid cells. (p_x, p_y) denotes the shape of anchor priors.

To decrease the number of false alarms in the detection results of recall-priority branch, we replace the feature pyramid in the YOLO module with BiFPN. This is a combination of efficient bidirectional cross-scale connections and weighted feature fusion [34]. Such connections are improved from PANet [36]. However, unlike PANet, nodes with only one input are removed. In addition, extra connections are added on each level from original input to output node, and each bidirectional path is treated as a feature layer in BiFPN. More high-level feature fusion is ensured by repeating the same layer multiple times. Fast normalized fusion is selected as weighted feature fusion, whose function is $O = \sum_i I_i w_i / (\in + \sum_j w_j)$. In this function, $w_i \leq 0$, and $\in = 0.0001$ is a value to keep stability. Such fusion is much more efficient than softmax-based fusion. The structure of precision-priority branch is shown in Figure 6.

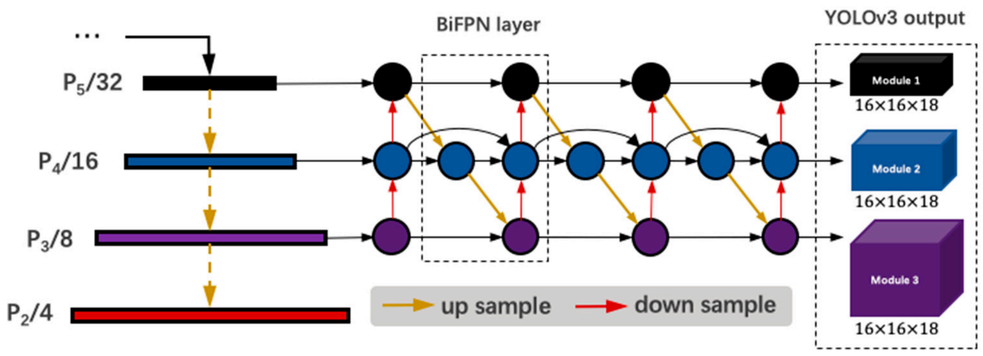

Figure 6. The feature map produced by the backbone first go through BiFPN layers to integrate features further. Finally, the output modules from YOLOv3 give the predicted boxes based on anchor priors.

2.3. Priority-Based Selection

Priority-based selection serves as a filter to distinguish accurate boxes from the boxes from precision-priority branch (B_p) and the boxes from recall-priority branch (B_R).

The strategy to filter boxes changes with whether they overlap:

1. If B_P has a high overlap with B_R, we reserve B_R.
2. If a box has low or no overlap with all boxes from the other branch, and its confidence is higher than the threshold (the threshold is less strict for precision-priority branch), we reserve it.

Strategy (1) works in the area where B_P and B_R reach a consensus that ships exist, but they produce different boxes. At this time, B_R is selected because the boxes from recall-priority can promise the recall for side-by-side ships. Strategy (2) deals with disagreements between the two branches. Boxes with low confidence are filtered and more boxes from the precision-priority branch are reserved because it is less likely to sound false alarms. Strategy (2) not only reduces the number of false alarms, but also keeps high potential boxes. This selection algorithm is shown in Figure 7.

```
Input: $R=\{r_1,\ldots,r_m\}$, $C_r=\{C_{r1},\ldots,C_{rm}\}$, $\theta_r$,
       $P=\{p_1,\ldots,p_n\}$, $C_p=\{C_{p1},\ldots,C_{pn}\}$, $\theta_p$, $\theta_{IoU}$
Output: $B$
$B=\{\}$, $P_{invalid}=\{\}$
while $R$ is not empty do
    $r_j = R[0]$
    if $\exists p_i$, $IoU(p_i,r_j) \geq \theta_{IoU}$ then
        $B = B \bigcup r_j$, $P_{invalid} = P_{invalid} \bigcup P_i$
    else
        if $C_{rj} \geq \theta_r$ then
            $B = B \bigcup r_j$
    $R = R - r_j$
for $p_i$ in $(P - P_{invalid})$ do
    if $C_{pi} \geq \theta_p$ then
        $B = B \bigcup p_i$
return $B$
```

Figure 7. R and P are the list of detection boxes produced by the two branches. C_r and C_p contain corresponding detection confidence. If boxes have overlap higher than θ_{IoU}, the box from the precision-priority branch will be eliminated. The confidence thresholds for the two branches are defined in the form of θ_r, θ_p. The accuracy of detection results is the highest by setting $\theta_{IoU} = 0.3$, $\theta_r = 0.4$, and $\theta_p = 0.8$ in our work.

PBS works as an efficient arbiter, ensuring high-quality boxes. The experiments described in the next section show that it leads to noticeable improvement compared with detection results from each branch.

3. Experiments and Results

To evaluate the performance of our proposed single-shot ship detection method, we compared it with other methods. The settings of the experiments introduced in this section include datasets, evaluation metrics, and compared methods. The efficiency of our method will also be presented.

3.1. Dataset

The HRSC2016 dataset contains images from two scenarios, including ships at sea and ships near the shore [24]. These 1061 images, including 2976 samples, were selected from Google Earth, and consist of images from six famous ports: Murmansk, Everett, Newport, Rhode Island, Mayport Naval Base, Norfolk Naval Base, and San Diego Naval Base. Their sizes range from 300 × 300 and 1500 × 900, while the resolutions vary from 2 m to 0.4 m. However, by studying the official images and annotations, the standard for annotations was found to be inconsistent. The two aspects to the problem are small ships and incomplete ships. In some images, they are annotated, but most are ignored by the author. Sample images and official annotated boxes are shown in Figure 8.

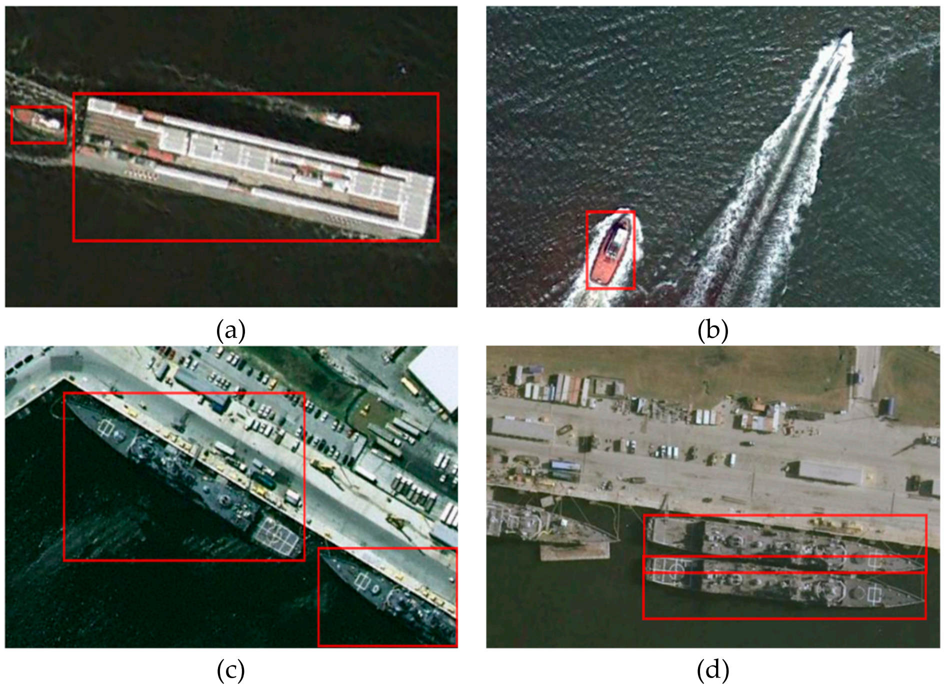

Figure 8. Sample images and official annotated boxes. The inconsistent standards for small ships and incomplete ships are very noticeable.

To improve the dataset, we selected 360 more images from Google Earth to augment it and re-annotated these images. The resolutions of the images were between 0.27 m to 2.15 m. Their sizes range from 450 × 350 to 1200 × 850, and most of them are smaller than 1000 × 600. Like the original images in HRSC2016, most of the added samples are on the sea or near the shore. Sample images in our dataset are shown in Figure 9.

We divided all the samples into large, small, and incomplete. Varying from the method of Lin et al. [37], we labeled complete ships with bounding box area larger than 96 × 96 pixels as large ships, and other complete ones as small ships. The improved dataset includes a total of 1421 images and 5058 samples, as shown in Table 1.

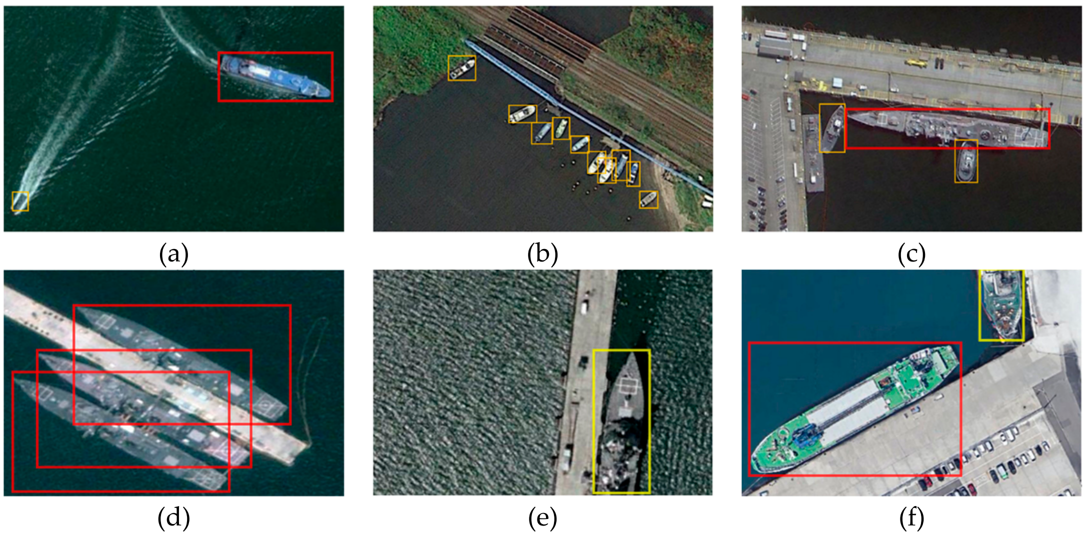

Figure 9. Samples of augmented images and their new annotations based on levels. Red boxes represent large ships, orange boxes represent small ships, and yellow boxes represent incomplete ships.

Table 1. Number of samples in three sets

Levels	**Number of Samples**		
	Training Set	**Validation Set**	**Testing Set**
Large	2204	285	1059
Small	825	70	333
Incomplete	181	11	90
Total	3210	366	1482

3.2. Evaluation Metrics

To measure the performance of our proposed methods quantitatively, we adopted precision, recall, and the precision–recall curve (PRC). The value of recall and precision can be calculated by TPs (true positives), FPs (false positives), and FNs (false negatives). If the IoU between a predicted box and ground truth is higher than 0.5, it is defined as TP. If the IoU is lower than 0.5, it is defined as FP. An actual object missed by detection is called an FN. Precision stands for the ratio of correct boxes and all predicted ones, and recall is the proportion of predicted correct boxes in all ground truth. AP, which is the area under PRC, is a comprehensive metric for avoiding the situation where recall and precision are unbalanced. The value of AP is between 0 and 1. The calculation methods are

$$Precision = \frac{TP}{TP+FP} \tag{18}$$

$$Recall = \frac{TP}{TP+FN} \tag{19}$$

3.3. Compared Methods

The detections we compare with consist of one-stage and two-stage frameworks. One-stage detections include YOLOv3 [32], RetinaNet [31], and FSAF [38]. Faster R-CNN [19] is the two-stage framework we use.

YOLOv3 is known for its efficiency. Referring to the feature pyramid structure and forming a multi-scale detection, it is good at detecting small objects. For better performance, we use darknet-53 with official pretrained weights as the backbone.

Using focal loss, RetinaNet is able to challenge the accuracy of two-stage detections after overcoming the problem of category imbalance. Images with a size of 640 × 640 go through ResNet-50 for feature extraction.

Like our proposed methods, FSAF also has an anchor-based branch and an anchor-free one. The best feature map is selected automatically by feature levels. Finally, predicted boxes produced by the two branches need to go through NMS. The base network of FSAF is also ResNet-50.

As a two-stage framework, Faster R-CNN applies a region proposal network to obtain candidate regions containing objects. The Fast R-CNN detector then classifies these regions. They share ResNet-50 as the backbone. Faster R-CNN is a popular method because of its high accuracy in detection tasks.

We adopted pretrained ResNet-50 on the Microsoft COCO dataset for RetinaNet, FSAF, and Faster R-CNN. For all methods, we used the Adam [39] optimization algorithm during 200 epochs of training. The initial learning rate remains 10^{-4} for the first 80 epochs and decays to 10^{-5} and 10^{-6} at epoch 80 and epoch 150. All networks are trained and tested in our dataset.

Considering that they all use NMS to merge results, networks with soft-NMS are also used.

3.4. Implementation Details

Data augmentation was applied before training the proposed network and the compared methods. It includes random flip respect to x-axis, shift augmentation, rescale 0.5, and rotate 90, 180, 270. In addition, more negative samples were added artificially to some images to improve the balance of positive and negative samples.

In the proposed single-shot detector, input images were resized to 512 × 512. Since NMS is not applied in the anchor-free recall-priority branch, boxes are selected if the confidence is higher than 0.3. The threshold of NMS in our anchor-based branch is 0.4. During training, the backbone is initialized by the weights of pretrained DLA-34 on the ImageNet database [40]. We follow Zhu et al. [38] to use $L = L_R + \lambda L_P$ as total optimization loss. The entire training process lasts for 200 epochs. The learning rate is set to 1.25×10^{-4} for the first 90 epochs and becomes 10 times smaller at epoch 90 and 150. The Adam optimizer is also used during the training. Our detector performs best after we trained it in this way. Our proposed method is implemented on the PyTorch framework and employs an Nvidia GeForce RTX 2080 Ti GPU with 11 GB memory for training.

3.5. Results

The comparison between our proposed network and other methods is presented both visually and quantitatively. The PRCs of five networks are shown in Figure 10, and the specific values of AP are compared in Table 2.

Table 2. Recall, precision, and AP values of methods on the same dataset

Methods	**P & R (%)**		**AP Values (%)**			
	P	**R**	**AP**	**AP$_L$**	**AP$_S$**	**AP$_{In}$**
Faster R-CNN	90.50	89.95	86.95	94.14	69.94	59.46
YOLOv3	93.24	83.81	83.27	92.73	58.47	52.45
RetinaNet	87.90	88.19	86.28	94.55	66.43	56.06
FSAF	91.55	91.36	89.71	96.12	76.23	53.32
Proposed	94.02	96.02	95.57	99.42	83.14	78.51

From Figure 10, it is obvious that our method transcends all other networks in both precision and recall. Specifically, the recall is much higher than others. As a two-stage detector, Faster R-CNN achieves good results in recall because the region proposal network produces proper boxes for candidate ships. YOLOv3 has the best performance in precision but has the lowest recall of ships of the compared methods. By observing the visual outputs of YOLOv3, we find that some ships are selected by bounding boxes, but the size and location of boxes are not precise enough, thereby making recall the lowest. In contrast, the recall of RetinaNet is high but the precision is unsatisfactory. FSAF has the steadiest performance.

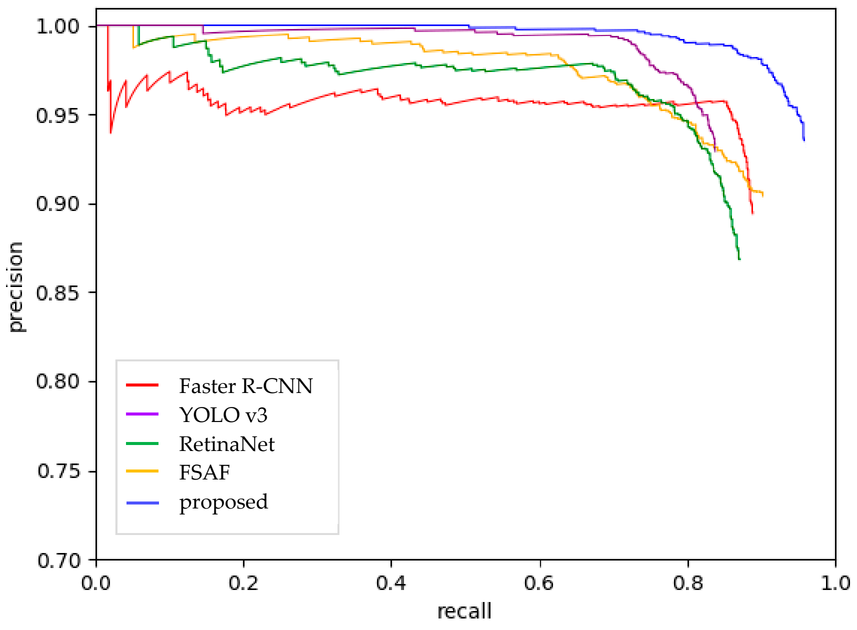

Figure 10. PRC of different methods. The proposed network shows significant improvement in both precision and recall.

Table 2 shows that our method achieves the best AP value. In addition to its nearly 100% AP of large ships, it makes strong progress in detecting small ships and ships with incomplete forms. Among all compared networks, the AP of FSAF is closest to our performance, especially in detecting small ships. Using the same pretrained weights with Faster R-CNN and RetinaNet, we can speculate that the combination of anchor-free and anchor-based modules works well in the task of detecting ships. Compared with FSAF, the heat map method in our anchor-free module and the feature extraction architecture in the anchor-based module achieve 5.86%, 3.30%, 6.91%, and 25.19% performance gains in AP, AP_L, AP_S, and AP_{In} respectively. The other three networks perform well in detecting large ships but are poor at detecting small ships and those with incomplete forms.

To further show the advantages of our proposed method, we list the number of false alarms and missed ships in Table 3. Surprisingly, YOLOv3 produces the same number of false alarms as the proposed network, despite the fact that it misses many more ships. At the early stages of our experiment, we found that YOLOv3 is good at accurate detection, so we used the YOLO output part as the main module in our precision-priority branch. From our observation of all the missed ships, we found that ships with different sizes were responsible for the most obstacles for improving recall value. By further fusing features, our method significantly reduced the number of obstacles. Meanwhile, our method was highly efficient in detecting side-by-side ships, reducing the number to zero, compared to tens of missed ships by other methods. Figure 11 shows the number of missed side-by-side ships after replacing NMS with soft-NMS in compared networks. Although soft-NMS is reported to significantly improve NMS [22], the number is only reduced by a small amount. Therefore, we believe that the key to solving this problem is to find center points more precisely instead of improving the performance of NMS. It is by this particular method that we can solve the problem so completely.

Table 3. Number of missed ships and false alarms of methods

Methods	False Alarms	Missed Ships		
		Side-by-Side Ships	Incomplete Ships	Multi-Scale Ships
Faster R-CNN	140	17	34	98
YOLOv3	90	36	41	177
RetinaNet	180	23	36	116
FSAF	125	25	38	65
Proposed	90	0	15	44

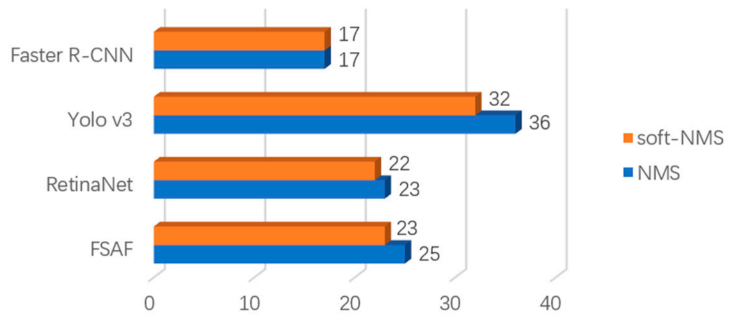

Figure 11. Number of missed side-by-side ships. Soft-NMS improves a small degree when conducting this task.

The good performance of our detection method in both precision and recall can be attributed to PBS combining outputs from two branches. Table 4 clearly shows that the recall-priority branch is good at getting high recall while the other branch produces bounding boxes much more accurately. Because of PBS, the recall rises little compared to the recall-priority branch and the precision of the filtered results nearly reaches that of the precision-priority branch. It is easy to explain why the recall gets higher: more boxes mean more chances to hit the targets. Since we keep some results from the recall-priority branch, it is inevitable that more false alarms will occur and produce a lower precision. To attain higher precision, a recall-priority branch with better precision is needed. This is one of the goals of our future work.

Table 4. Change of recall and precision after being filtered by PBS

Modules	Precision (%)	Recall (%)
Precision-Priority Branch	94.68	86.17
Recall-Priority Branch	90.11	95.48
PBS	94.02	96.02

Qualitative comparisons to other methods are shown in Figure 12. Compared with other methods, our networks achieve more robust detection of ship-like objects, side-by-side ships, and multi-scale ships. Our detector also shows favorable performance for incomplete ships.

The final efficiency analysis of networks is shown in Table 5. The inference speed of the proposed method is second only to YOLOv3. Considering that it is much better in precision and recall values, ranking second in speed is acceptable.

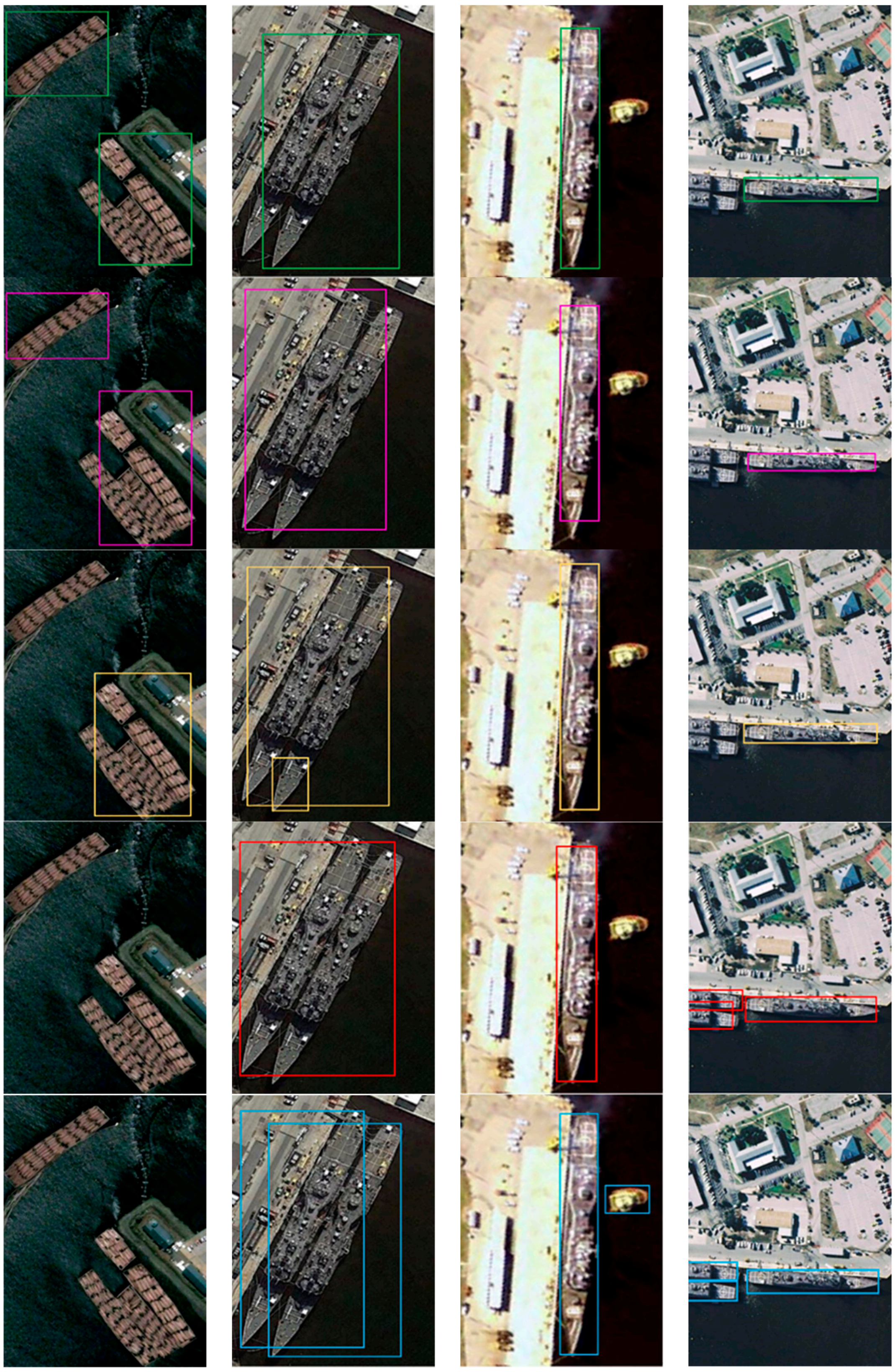

Figure 12. Visual comparisons of detection results in different methods. From top to bottom, each row is Faster R-CNN, YOLOv3, RetinaNet, FSAF, and the proposed method, respectively.

Table 5. Average inference time of all methods

Methods	Pretrained Backbone	Frames per Second
Faster R-CNN	ResNet-50	20.2
YOLOv3	Darknet-53	75.1
RetinaNet	ResNet-50	33.5
FSAF	ResNet-50	43.0
Proposed	DLA-34	55.8

4. Conclusions

In this paper, we find that many of the obstacles of ship detection are caused by side-by-side ships, shape-like objects, and multi-scale ships. First, to reduce the number of false alarms and missed ships, we split the task into two branches: the recall-priority branch without NMS, which improves recall; and the precision-priority branch, which is good at detecting ships precisely. The final bounding boxes are selected from the results of two branches by PBS. Next, we constructed a dataset based on HRSC2016 with more samples and stricter annotation standards.

Through experiments on our dataset, we can make three conclusions. (1) The bottle neck to reducing the number of times side-by-side ships were missed is not the quality of the suppression algorithm, but the accuracy of detected center points of ships. Accordingly, the recall-priority branch improves recall to a high level. In addition, false alarms can be avoided to an extent by better fusion of features in our precision-priority branch. (2) Through the use of PBS, the combination of different branches works well, and the AP values increase sharply compared to other networks. (3) The inference time of our proposed method is short enough to meet the requirements of real-time detection (above 30 fps). In the future, we aim to simplify the network and put it on an ASIC chip.

Author Contributions: Conceptualization, Q.W.; Methodology, Y.Z.; Dataset, W.S. and J.J.; Software, Y.Z.; Validation, Q.W. and N.J.; Formal Analysis, J.J. and W.S.; Writing—Original Draft Preparation, Y.Z.; Writing—Review and Editing, N.J. and Q.W.; Supervision, Z.M.; Funding acquisition, Z.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (61176037).

Acknowledgments: The authors sincerely thank Dong G. C. and Shen F. Y. for their suggestions. The authors also want to thank You Y.L. and LetPub for their language polish of the writing.

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

References

1. Li, Y.; Martinis, S.; Wieland, M. Urban flood mapping with an active self-learning convolutional neural network based on TerraSAR-X intensity and interferometric coherence. *ISPRS J. Photogramm. Remote Sens.* **2019**, *152*, 178–191. [CrossRef]
2. Sun, S.; Li, C.; Chee, P.W.; Paterson, A.H.; Jiang, Y.; Xu, R.; Shehzad, T. Three-dimensional photogrammetric mapping of cotton bolls in situ based on point cloud segmentation and clustering. *ISPRS J. Photogramm. Remote Sens.* **2020**, *160*, 195–207. [CrossRef]
3. Dong, C.; Liu, J.; Xu, F. Ship detection in optical remote sensing images based on saliency and a rotation-invariant descriptor. *Remote Sens.* **2018**, *10*, 400. [CrossRef]
4. Barnum, J. Ship detection with high-resolution HF skywave radar. *IEEE J. Ocean. Eng.* **1986**, *11*, 196–209. [CrossRef]
5. Tian, Y.; Wang, C.; Zhang, H. Spaceborne SAR Ship Detection and its Application on Marine fisheries monitoring. *Remote Technol. Appl.* **2007**, *22*, 503–512.
6. Novak, L.M.; Halversen, S.D.; Owirka, G.; Hiett, M. Effects of polarization and resolution on SAR ATR. *IEEE Trans. Aerosp. Electron. Syst.* **1997**, *33*, 102–116. [CrossRef]
7. Vachon, P.W.; Adlakha, P.; Edel, H.; Henschel, M.; Ramsay, B.; Flett, D.; Thomas, S. Canadian progress toward marine and coastal applications of synthetic aperture radar. *Johns Hopkins APL Tech. Digest.* **2000**, *21*, 33–40.

8. Kaplan, L.M. Improved SAR target detection via extended fractal features. *IEEE Trans. Aerosp. Electron. Syst.* **2001**, *37*, 436–451. [CrossRef]
9. Copeland, A.C.; Ravichandran, G.; Trivedi, M.M. Localized Radon transform-based detection of ship wakes in SAR images. *IEEE Trans. Geosci. Remote Sens.* **1995**, *33*, 35–45. [CrossRef]
10. Proia, N.; Pagé, V. Characterization of a Bayesian ship detection method in optical satellite images. *IEEE Geosci. Remote Sens. Lett.* **2009**, *7*, 226–230. [CrossRef]
11. Shu, C.; Ding, X.; Fang, C. Histogram of the oriented gradient for face recognition. *Tsinghua Sci. Technol.* **2011**, *16*, 216–224. [CrossRef]
12. Liu, G.; Zhang, Y.; Zheng, X.; Sun, X.; Fu, K.; Wang, H. A new method on inshore ship detection in high-resolution satellite images using shape and context information. *IEEE Geosci. Remote. Sens. Lett.* **2013**, *11*, 617–621. [CrossRef]
13. Yu, Y.; Guan, H.; Li, D.; Gu, T.; Tang, E.; Li, A. Orientation guided anchoring for geospatial object detection from remote sensing imagery. *ISPRS J. Photogramm. Remote. Sens.* **2020**, *160*, 67–82. [CrossRef]
14. Lin, H.; Shi, Z.; Zou, Z. Fully convolutional network with task partitioning for inshore ship detection in optical remote sensing images. *IEEE Geosci. Remote. Sens. Lett.* **2017**, *14*, 1665–1669. [CrossRef]
15. Xie, X.; Li, B.; Wei, X. Ship Detection in Multispectral Satellite Images Under Complex Environment. *Remote Sens.* **2020**, *12*, 792. [CrossRef]
16. Chen, Y.; Li, Y.; Wang, J.; Chen, W.; Zhang, X. Remote Sensing Image Ship Detection under Complex Sea Conditions Based on Deep Semantic Segmentation. *Remote Sens.* **2020**, *12*, 625. [CrossRef]
17. Dechesne, C.; Lefèvre, S.; Vadaine, R.; Hajduch, G.; Fablet, R. Ship Identification and Characterization in Sentinel-1 SAR Images with Multi-Task Deep Learning. *Remote Sens.* **2019**, *11*, 2997. [CrossRef]
18. Rostami, M.; Kolouri, S.; Eaton, E.; Kim, K. Deep Transfer Learning for Few-Shot SAR Image Classification. *Remote Sens.* **2019**, *11*, 1374. [CrossRef]
19. 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]
20. Zhang, S.; Wu, R.; Xu, K.; Wang, J.; Sun, W. R-CNN-based ship detection from high resolution remote sensing imagery. *Remote Sens.* **2019**, *11*, 631. [CrossRef]
21. Zhang, Z.; Guo, W.; Zhu, S.; Yu, W. Toward arbitrary-oriented ship detection with rotated region proposal and discrimination networks. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 1745–1749. [CrossRef]
22. Bodla, N.; Singh, B.; Chellappa, R.; Davis, L.S. Soft-NMS–improving object detection with one line of code. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5561–5569.
23. Yu, F.; Wang, D.; Shelhamer, E.; Darrell, T. Deep layer aggregation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2403–2412.
24. LB, W. A high resolution optical satellite image dataset for ship recognition and some new baselines. In Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods, Porto, Portugal, 24–26 February 2017.
25. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.
26. Zhu, X.; Hu, H.; Lin, S.; Dai, J. Deformable convnets v2: More deformable, better results. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9308–9316.
27. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 764–773.
28. Cao, Z.; Simon, T.; Wei, S.E.; Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7291–7299.
29. Newell, A.; Yang, K.; Deng, J. Stacked hourglass networks for human pose estimation. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 483–499.
30. Law, H.; Deng, J. Cornernet: Detecting objects as paired keypoints. In Proceedings of the 2018 European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 734–750.
31. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.

32. Redmon, J.; Farhadi, A. Yolov3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
33. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
34. Tan, M.; Pang, R.; Le, Q.V. Efficientdet: Scalable and Efficient Object Detection. *arXiv* **2019**, arXiv:1911.09070.
35. Zhuang, S.; Wang, P.; Jiang, B.; Wang, G.; Wang, C. A Single Shot Framework with Multi-Scale Feature Fusion for Geospatial Object Detection. *Remote Sens.* **2019**, *11*, 594. [CrossRef]
36. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
37. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the 2014 European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 740–755.
38. Zhu, C.; He, Y.; Savvides, M. Feature selective anchor-free module for single-shot object detection. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 840–849.
39. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
40. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, USA, 20–25 June 2009; pp. 248–255.