LSDNet: A Lightweight Ship Detection Network with Improved YOLOv7

Cui Lang
Harbin Normal University
Xiaoyan Yu
yuxiaoyan@hrbnu.edu.cn
Harbin Normal University
Xianwei Rong
Harbin Normal University

Research Article

Keywords: Ship detection, Convolution neural network, PConv, GhostConv, Mosaic-9

Posted Date: November 9th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3572198/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Journal of Real-Time Image Processing on March 27th, 2024. See the published version at https://doi.org/10.1007/s11554-024-01441-9.

Abstract

Accurate ship detection is critical for maritime transportation security. Current deep learning-based object detection algorithms have made marked progress in detection accuracy. However, these models are too heavy to be applied in mobile or embedded devices with limited resources. Thus, this paper proposes a lightweight convolutional neural network shortened as LSDNet for mobile ship detection. In the proposed model, we introduce Partial Convolution (PConv) into YOLOv7-tiny to reduce its parameter and computational complexity. Meanwhile, GhostConv is introduced to further achieve lightweight structure and improve detection performance. In addition, we have replaced Mosaic-4 data augmentation with Mosaic-9 to enhance the robustness of the model. We compared the proposed LSDNet with other approaches on a publicly available ship dataset, SeaShips7000. The experimental results show that LSDNet achieves higher accuracy than other models with less computational cost and parameters. The test results also suggest that the proposed model can meet the requirements of real-time applications.

1. Introduction

Due to the increasing frequency of marine activities and the growing number of ships on the sea, ship detection is particularly important for enhancing maritime safety. Although existing marine monitoring systems have achieved marked progress in detecting ships such as automatic identification systems (AIS) [1] and video surveillance systems, just some large ships have installed AIS, which may result in missing inspection of small ships. Video surveillance systems generally require long-term manual observation, easily causing visual fatigue. This means that more advanced and powerful methods are needed for ship detection.

With the development of deep learning technology, many advanced ship detection methods have emerged, providing strong support for offshore ship monitoring. Unlike traditional object detection methods, deep learning based on object detection algorithms [2] have low dependence on manual labor and can automatically extract useful features for end-to-end detection. According to processing flow of algorithms, deep learning-based algorithms can be classified into two categories: two-stage algorithms and single-stage algorithms.

The two-stage algorithm divides the detection problem into two steps: the first is to extract the object region, and the second is to classify and recognize the region using CNN. Such type of algorithms mainly includes R-CNN [3], Fast R-CNN [4], and Fast R-CNN [5]. Although their error rate and missed detection rate are low, its detection speed is slow and cannot meet the real-time monitoring scenarios. Single stage object detection algorithms only require one feature extraction to achieve object detection, which is faster than two-stage object detection algorithms. These algorithms mainly include SSD [6], YOLO [7-14] series. Although the computational burden of a single stage target detection algorithm is much lower than that of a two-stage target detection algorithm, it is still not lightweight enough to be applied to resource limited offshore monitoring systems.

To address this issue, many lightweight CNN network models have emerged, such as MobileNetV1/V2/V3 [15-17], ShuffleNetV1/V2 [18-19], MobileOne [20], etc. These networks can reduce the number of parameters and computational complexity, while most of them generally leverage the detection accuracy. To tackle this problem, we propose an improved ship detection model based on YOLOv7-tiny. The proposed model can achieve good detection results while remaining its lightweight, so it is suitable for the offshore monitoring systems with limited resources. The contributions of this study can be summarized as follows:

1. We propose a lightweight network shorted as LSDNet for real-time monitoring of offshore vessels.
2. Mosaic-9 data augmentation is introduced to enhance our model's robustness.
3. Extensive experiments are conducted on a large ship dataset named SeaShips7000 and the results has proven the effectiveness and efficiency of the proposed model.

The rest of this article is organized as follows: Section 2 briefly reviews the existing YOLO algorithm. Section 3 describes the proposed lightweight network (LSDNet) for ship detection. Section 4 presents extensive experimental results on SeaShips7000 dataset. Finally, we summarize this work in Section 5.

2. Related works

2.1 The YOLO Series

In 2015, Joseph Redmon et al. proposed a one-stage object detection model named as You Only Look Once (YOLO) [7] to realize an end-to-end object detection. After that, the successors of YOLO were gradually proposed for improved performance. Compared with

YOLOv1, YOLOv2 [8] adds a batch normalization layer after all the convolutional layers to accelerate convergence and reduce overfitting. Also, K-means clustering technology [21] is used for the model training to predict more accurate boundary boxes. To reduce the computation cost of the model, 1 × 1 convolution is used for dimensionality reduction. In 2018, YOLOv3 [9] was proposed by Joseph Redmon and Ali Farhad, which uses DarkNet-53 as the backbone and introduces a feature pyramid structure to achieve more competitive detection accuracy with fewer model parameters. To reduce the computational complexity of the model while maintaining its comparable accuracy to YOLOv3, YOLOv4 [10] was proposed in April 2020 by Alexey Bochkovskiy et al., in which CSP (Cross Stage Partial Connection)-Darknet-53 replaces the DarkNet-53 [22]. Also, the SPP [23] module with pooling kernels of different scales is used to obtain spatial features of different receptive fields. A few months later, YOLOv5 [11] was released by Glenn Jocher, and it was no longer developed using Darknet and instead Python. In the YOLOv5, multiple data augmentation technologies are adopted at the input end to enhance its robustness, and C3 modules are used in the backbone network to make the model lightweight. In addition, the network combines a fast spatial pyramid structure and a path aggregation network (PANet) [24] to effectively enhance its feature fusion ability. Afterwards, many versions were released successively such as YOLOR [12] and YOLOX [13], until the release of YOLOv7 [14] in July 2022.

YOLOv7 uses efficient long-range attention network (ELAN) with infinitely stacked computational blocks to improve its detection accuracy while remaining its fast inference speed. ELAN is a strategy that allows deep models to learn and converge more effectively by controlling the shortest and longest gradient paths. E-ELAN enhances the learning ability of the network by combining the features of different groups via shuffling and merging cardinality, without damaging the original gradient path. This exploration is based on the small model YOLOv7-tiny.

2.2 PConv

In 2023, Jierun Chen et al. proposed a simple Partial Convolution (PConv) [25]. PConv utilizes the high similarity of feature maps between different channels, which merely uses conventional Conv on some input channels for spatial feature extraction while keeping the remaining channels unchanged. As shown in Figure 1, the purple part in the output is a channel that has undergone conventional convolution operations, while the remainders have no such operations. Thus, it can markedly reduce computation cost and internal memory access. The FLOPs of PConv are:

$$h \times w \times k^2 \times c_p^2$$

Also, the memory access of PConv is relatively small, namely:

$$h \times w \times k^2 \times c_p^2$$

where *w* and h denotes the width and height, $h \times w \times k^2 \times c_p^2$ is the number of channels.

PConv solves the problem of frequent memory access such as DWConv and GConv, and remains high FLOPS while reducing FLOPs, which is beneficial for reducing latency.

$$h \times w \times k^2 \times c_p^2 \quad (1)$$

where FLOPs denotes the number of floating-point operations, and FLOPS represents the number of floating-point operations per second.

2.3 GhostConv

The conventional feature extraction methods use multiple convolutional kernels to perform convolutional mapping operations on all channels in the input feature maps. However, in the deep neural networks, stacking a large number of convolutional layers requires huge parameters and heavy computation, and also generates many rich or even redundant feature maps. Thus, several model compression methods have emerged such as pruning [26], quantization [27], knowledge distillation [28]. Although these methods can markedly reduce the number of parameters, they generally endure some issues such as complex model design and training difficulties.

The GhostNet model [29] was proposed to reduce computational complexity. first extract the features from the input, and then further reduces the computational complexity via inexpensive operations to generate new feature maps. Finally, the two output feature maps are concatenated to obtain the final results. This greatly reduces the computation cost of non-critical features with cheap operations,

decreasing the requirements for computing resources without leveraging the model's performance. The structural diagram of the GhostConv module is shown in Figure 2.

2.4 Mosaic-9

The Mosaic-4 method is an extension of the CutMix [30], while it does not overlap and fuse two images, instead uses four images for cropping and stitching to generate a new image. This can effectively enrich the target background and prevent a decrease in generalization ability due to similar backgrounds during training. Mosaic-9 [31] improves the Mosaic-4 method via adding a channel with nine images on the top of the original channel and using three channels for feature enhancement. This makes the scales of features from the training dataset more diverse and weakens the interference of the background on the target features, enhancing the network's generalization ability. The structural diagram of Mosaic-9 is shown in Figure 3, where m1 represents the original single image, m4 denotes four images for cropping and stitching, and m9 nine images for cropping and stitching.

3. The proposed Ship Detection Framework

To solve the challenge in deploying ship detection networks on mobile monitoring systems, we propose a lightweight ship detection network based on YOLOv7-tiny. The architecture of the proposed network is shown Figure 4. We can see that the proposed LSDNet includes two main modules, improved Backbone and Head. The improved Backbone network aims to extract more discriminative ship features while reducing computation cost and mode size. In this section, we provide a detailed description of the improved components in the Backbone and Head networks.

3.1 Improvement Areas

To enable the existing YOLOv7-tiny model to be applied in mobile devices with limited resources, it is essential to further make the model lightweight. Thus, we made the following improvement on the YOLOv7-tiny to reduce its parameters and computational complexity while reserving its good detection accuracy.

(1) The convolutional blocks are replaced by PConv with a kernel size of 3×3 and a step size of 1 in the MCB module to reduce computational bottlenecks and memory overhead.

(2) We use GhostConv to replace the Conv convolution block in the network and the last convolution in the MCB module, ensuring model's detection accuracy while further reducing its weight. The improved MCB module is named as Light-MCB.

(3) To enhance the robustness of the model and improve detection accuracy, we also replaced Mosaic-4 with Mosaic-9 data augmentation.

Since the more suitable anchor boxes generally enable the network easier to learn, we use K-means clustering to automatically generate anchor boxes based on the dataset, rather than using YOLOv7-tiny's original anchors.

3.2 Light-MCB

The original MCB module is shown in Figure 5 (a), including two branches. One branch undergoes three convolutions, while the other branch only undergoes one convolution. Subsequently, the feature maps obtained from each convolution operation are concatenated and then output by the convolution. To achieve a better cost-effective performance, we proposed an improved MCB module shorted as Light-MCB. The improved MCB module is shown in Figure 5 (b). We can see that a branch first undergoes a convolution with a kernel size of 1 × 1 and a step size of 1, and then undergoes two PConvs, while the other branch is a convolution with a kernel size of 1 × 1 and a step size of 1. Next, the output feature maps after standard convolution and PConv operation will be concatenated, followed by a GhostConv.

4. Experimental results and analysis

To evaluate the performance of our proposed network, we conducted extensive experiments on the benchmark dataset Seaships7000. We compared the proposed LSDNet with other networks via quantitative and qualitative results. The implementation details and experimental results are described as follows.

4.1 Implementation Details

4.1.1 Experimental Environment and Parameter Settings

The experimental environment is based on Pycharm software with operating system Ubuntu 22.04, deep learning framework PyTorch1.13, programming language Python 3.9, and integrated development environment CUDA11.4. The CPU model is 12th Gen Intel (R) Core (TM) i5-12490F, with a main frequency of 3 GHz, and the graphics card is NVIDIA GeForce GTX 1080Ti. For the optimal hyperparameters used in our network, the basic learning rate, momentum, and weight attenuation are set to 0.01, 0.937, and 0.0005, respectively, with the optimizer Adam. In all experiments, the size of the input image is 640 × 640 pixels, epoch is set to 200, and batch size is set to 32. The other parameters use the default values from the original YOLOv7-tiny.

4.1.2 Evaluating Indicator

The nine objective metrics are used to evaluate the effectiveness and efficiency of the proposed model, namely Precision, Recall, F1, mAP@0.5, mAP@0.5:0.95, Inference, Param, FLOPs, Train time.

Precision is used to evaluate the accuracy of ship predictions, reflecting the proportion of actual positive samples among all predicted positive samples. Recall is used to evaluate whether all ships in the test dataset have been correctly predicted, which reflects the proportion of positive samples correctly predicted by the model to the total positive samples. The F1 score is the harmonic average of Precision and Recall mAP@0.5 and mAP@0.5:0.95 is a comprehensive indicator for measuring the accuracy and robustness of ship detection. Param is the parameter quantity that reflects the memory usage of the model, and FLOPs is the computational cost that reflects the complexity of the model. The fewer model parameters and FLOPs, the lower the cost of detecting the model. In addition to the above evaluation indicators, we also provided inference speed and training time to verify the progressiveness of our model.

4.2 Dataset Settings

4.2.1 Dataset

We used the free and publicly available ship dataset SeaShips7000 [31] as the dataset for this experiment, which consists of a total of 7000 images and includes six types of ships: ore ships, bulk cargo ships, regular container ships, container ships, fishing ships, and passenger ships. In this experiment, they are randomly divided into a training set, a validation set, and a testing set, with a ratio of 3:1:1.

Figure.6 shows a statistical analysis on the number of samples labeled with the type of ships in the dataset, as well as the size and shape of the labeled boxes. As shown in Figure. 6(a), uneven distribution of the numbers for six types of ships increases the difficulty of detection tasks. Figure 6 (b) reveals that the value of the aspect ratio of the candidate boxes is generally large, conforming to the shape characteristics of the ship itself. In Figure 6 (c), x and y represent the ratio of the coordinate of the center point of the annotation box to the size of the instance, respectively. Fig.1 (d) shows that the SeaShips7000 dataset has a wide range of ship instances and is concentrated in the middle and bottom positions of an image.

4.2.2 K-means

In detection tasks, appropriate anchor boxes are beneficial for improving detection accuracy and speed, while the anchor boxes in YOLOv7 tiny are based on the COCO dataset. To make the training more suitable for our dataset, we used K-means clustering [21] to automatically generate anchor boxes during the training process. K-means is essentially a data partitioning method based on Euclidean distances, where dimensions with high mean and variance have a decisive impact on the clustering results of the data. The core goal is to divide the given dataset into K clusters (where K is a hyperparameter) and provide the corresponding center point for each sample. The core part is to first fix the center point, adjust the category of each sample to reduce the loss function, then fix the category of each sample, adjust the center point, and continue to reduce the value of the loss function. These two processes cycle alternately until the loss function monotonically decreases to the minimum (minimum) value.

The size comparison between the anchor box generated by K-means clustering and the original anchor box is shown in Table 1. We can see that compared with the original anchors, the size of clustered anchor is larger and is more consistent with the aspect ratio of ships.

Table 1 Comparison of Anchor Frames.

anchors	Big	Medium	Small
original	116 90	30 61	10 13
	156 198	62 45	16 30
	373 326	59 119	33 23
K-means [21]	412 49	142 41	54 13
	304 83	232 40	98 23
	506 87	237 59	186 25

We performed a performance comparison of models with original anchor boxes and the automatically generated anchor boxes, and the experimental results are shown in Table 2. YOLOv7-tiny* denotes the anchors generated by K-means clustering.

Table 2 Comparison of YOLOv7-tiny and YOLOv7-tiny*.

Methods	Precision	Recall	F1	mAP@0.5	mAP@0.5:0.95	Inference(ms)	Params(M)	FIOPs(G)	Train time(h)
YOLOv7-tiny [14]	0.970	0.958	0.964	0.983	0.750	1.8	6.02	13.1	3.118
YOLOv7-tiny*	**0.979**	**0.966**	**0.972**	**0.985**	**0.760**	1.9	6.02	13.1	3.110

From Table 2, we can see that the anchor boxes automatically generated using K-means clustering are more suitable for the dataset used in this experiment. Compared to the original YOLOv7-tiny, the accuracy of YOLOv7-tiny* is improved by 0.9%, and the recall rate is improved by 0.8%, mAP@0.5 gets an increase of 0.2%, and mAP@0.5 0.95 is increased by 1%. It can be derived that we can achieve better results using K-means clustering to generate anchor boxes, so the following experiments are all based on the K-means clustering.

4.3 **Training Results and Analysis**

To better evaluate the performance of the proposed LSDNet, we analyzed its training results. In Algorithm 1, we describe the data training process in detail. During training, we set the epoch to 200 and the loss curve of LSDNet is given in Figure 7. We can see that the loss value continuously decreases and converges after 200 epoches of training. The Precision and Recall curves rise slowly and converge rapidly, indicating that the model is well trained without overfitting.

Algorithm 1 Training Procedure for LSDNet

Input: *Original image of ship detection*

Output: *Calibration results of detection in lightweight conditions*

Initialize LSDNet D ^ θ with random weights θ.

Set the training stage: num_epochs=200, batch_size=16.

Prepare the normal dataset Seaships7000_trainval.

for *i in num_epochs* **do**

repeat

Take a batch images M from Seaships7000_trainval.

for *j in batch_size* **do**

if *random. randint (0, 3) > 0* **then**

Randomly select a batch of images and their corresponding labels from the training dataset.

Apply data augmentation techniques to the selected images.

Use the augmented images and labels to calculate the loss.

end if

end for

Update LSDNet D ^ θ according to detection loss.

until *all images have been fed into training models*

end for

We made quantitative comparison between the proposed LSDNet and the original YOLOv7-tiny on six types of ships in terms of accuracy, recall, and mAP. Additionally, the plotted P-R curves for each type of ships are shown in Figures 8 and 9. We can observe that the proposed model achieves the better performance on most types of ships in terms of accuracy, recall, mAP and overall mAP than the original YOLOv7-tiny. This proves the superiority of our network in ship detection.

The F1 score is an important indicator of model stability, and the higher its value, the better the model performance. We calculated the F1 values of YOLOv7-tiny and LSDNet, and plotted corresponding curves, as shown in Figure 10. It is obvious that our model obtains a F1 score greater than YOLOv7-tiny, indicating better performance of the LSDNet model.

The confusion matrices on the validation sets are shown in Figure 11 to further evaluate the detection performance of the trained model.

It is obvious that the proposed model achieves the FP and FN of most types of ships comparable or even better than the original data, indicating that the improved model has higher classification accuracy.

4.4 **Quantitaive Evaluation**

To verify the effectiveness of our proposed network, we made a quantitative evaluation on SeaShips7000 dataset for the YOLO series and other lightweight backbone networks. The YOLO series includes YOLOv5n, YOLOv7-tiny, and YOLOv7-tiny*. The popular lightweight networks, MobileNetv3 [17], ShuffleNetv2 [19], and MobileOne [20], replace the backbone of YOLOv7-tiny. The experimental results are shown in Figure 12, Table 3 and Table 4.

Figure 12 shows the mAP, Precision, and Recall curves of all detection algorithms during model training. It can be seen that our model has slightly higher mAP and recall values than other models, and maintains a high level of accuracy. All curves slowly rise and converge rapidly, indicating that the model has undergone good training and no overfitting has occurred.

Table 3 shows that our model achieved the best performance in the YOLO series, as its accuracy, recall, and mAP were improved compared to the original YOLOv7-tiny. In addition, our model is more lightweight, with a size of only 3.69MB, accounting for only 61.3% of YOLOv7-tiny. As shown in Table 4, our model outperforms other lightweight backbone networks in terms of accuracy, recall, and

mAP. Although it has slightly higher computational complexity than MobileNetv3, our model has the least number of parameters. The experimental results also show that our model has the fastest inference speed, indicating that our model can achieve real-time monitoring, thus its cost-effectiveness is better than other models.

Table 3 Performance Comparison on SeaShip7000 Dataset (Mainly in YOLO Series).

Methods	Precision	Recall	F1	mAP@0.5	mAP@0.5:0.95	Inference(ms)	Params(M)	FlOPs(G)	Train time(h)
YOLOv5n [11]	0.959	0.959	0.959	0.980	0.745	0.9	1.77	4.2	2.522
YOLOv7-tiny [14]	0.970	0.958	0.964	0.983	0.750	1.8	6.02	13.1	3.118
YOLOv7-tiny*	**0.979**	0.966	0.972	0.985	**0.760**	1.9	6.02	13.1	3.110
Ours	0.974	**0.970**	**0.972**	**0.985**	0.757	2.5	3.69	7.4	4.904

Table 4 Performance Comparison on SeaShip7000 Dataset (Mainly in Lightweight Backbone Series).

Methods	Precision	Recall	F1	mAP@0.5	mAP@0.5:0.95	Inference(ms)	Params(M)	FlOPs(G)	Train time(h)
MobileNetv3 [17]	0.969	0.946	0.957	0.981	0.723	4.1	4.18	6.9	3.129
ShuffleNetv2 [19]	0.967	0.943	0.955	0.979	0.717	2.9	4.51	8.6	3.272
MobileOne [20]	0.954	0.949	0.951	0.977	0.713	2.8	4.48	10.3	12.319
Ours	**0.974**	**0.970**	**0.972**	**0.985**	**0.757**	2.5	3.69	7.4	4.904

4.5 **Qualitative Evaluation**

In the qualitative evaluation, we selected the baseline model as a reference and presented the visual results of our model and baseline model, as shown in Figure 13. The first column in Figure 13 shows the original image, the second column shows the detection results of YOLOv7-tiny, and the third column shows the detection results of our model.

From Figure 14, it can be seen that our network has a much higher detection accuracy for occluded ships (the first row) than the original YOLOv7-tiny. Moreover, our network has higher detection accuracy in the hazy night (second line). Due to the small size of the fishing boat, YOLOv7-tiny mistakenly detected two fishing boats that were closer together as one (in the third row), and our model was able to effectively detect both fishing boats.

4.6 **Ablation Experiment**

To explore the effectiveness and efficiency of the proposed network, we performed an ablation study on the components of the LSDNet architecture and the experimental details are shown in Table 5. It can be seen that when only PConv is added, parameters and computation cost are reduced markedly, while the values of mAP@0.5:0.95 are decreased. When only GhostConv is added, mAP remains almost unchanged. Although the model parameters and computation cost has decreased, it is still not lightweight enough. When PConv and GhostConv are added together, the model parameters and computational complexity are further reduced, and mAP remains at a high level. In addition to PConv and GhostConv, our network structure also incorporates Mosaic-9 data augmentation, which can enhance the robustness of the model while maintaining its lightweight. It is obvious from Table 5 that the proposed LSDNet achieves the comparable performance to YOLOv7-tiny while requiring less parameters and computation cost. This indicates that LSDNet achieves a better balance between detection accuracy and model complexity.

Table 5 Comparison of Ship Detection Performance of Different Modules.

Number	K-means [21]	PConv [25]	GhostConv [29]	Mosaic-9 [31]	Params(M)	FlOPs(G)	mAP@0.5	mAP@0.5:0.95
N1					6.02	13.1	0.983	0.750
N2	√				6.02	13.1	0.985	**0.760**
N3	√	√			4.35	9.2	0.985	0.738
N4	√		√		5.80	12.3	0.985	0.759
N5	√	√	√		3.69	7.4	0.984	0.750
N6	√	√	√	√	**3.69**	**7.4**	**0.985**	0.757

5. Conclusions

In this study, we propose a lightweight CNN framework for the ship detection in maritime monitoring systems. The proposed LSDNet is more lightweight by introducing PConv and GhostConv to original YOLOv7-tiny, which is suitable to be applied in mobile devices with limited resources. Moreover, Mosaic-9 data augmentation supersedes the Mosaic-4 to further increase the diversity of training samples, enhancing the robustness of the proposed model and improving the detection accuracy of small ship targets. Also, extensive experiments on benchmark dataset SeaShips7000 reveal that our network can achieve good performance for various ship detection while keeping its lightweight. Compared with other ship detection methods, our approach achieves a better balance between model complexity and detection accuracy. Further work can be carried out from the following directions:

(1) The proposed network mainly performs lightweight processing, but the detection of small targets [33] still faces challenges, as small ships with long distances are difficult to detect. Accordingly, we can improve the detection of small targets by adding detection head for small ship targets.

(2) Most of existing ship detection models are based on single mode, which is hard to achieve satisfactory performance for ship detection under complex situations such as occluded ships. Thus, we will focus on multimodal [34] ship detection via incorporating with radar technology to improve ship detection performance.

References

1. Technical Characteristics for An Automatic Identification System Using Time-Division Multiple Access in the VHF Maritime Mobile Band, Standard ITU-R M.1371, Feb. 2014. [Online]. Available: http://www.itu.int/rec/R-REC-M.1371/en
2. Zou Z, Chen K, Shi Z, et al. Object detection in 20 years: A survey[J]. Proceedings of the IEEE, 2023.
3. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.
4. R. Girshick, “Fast R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1440–1448.
5. S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards realtime object detection with region proposal networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
6. W. Liu, D. Anguelov, and D. Erhan, “SSD: Single shot MultiBox detector,” in Proc. Eur. Conf. Comput. Vis., Oct. 2016, pp. 21–37.
7. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 779–788.
8. J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 7263–7271.
9. J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” 2018, arXiv:1804.02767.
10. A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal speed and accuracy of object detection,” 2020, arXiv:2004.10934.
11. J.Glenn,S.Alex,andB.Jirka.(2021).Ultralytics/YOLOv5:V6.0(Versionv6.0).[Online].Available:http://doi.org/10.5281/zenodo.63715
12. Wang C Y, Yeh I H, Liao H Y M. You only learn one representation: Unified network for multiple tasks[J]. arXiv preprint arXiv:2105.04206, 2021.
13. Ge Z, Liu S, Wang F, et al. Yolox: Exceeding yolo series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.

14. Wang C Y, Bochkovskiy A, Liao H Y M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 7464-7475.
15. Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv:1704.04861, 2017.
16. Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.
17. A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 1314–1324.
18. X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 6848–6856.
19. N. Ma and X. Zhang, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 122–138.
20. Vasu P K A, Gabriel J, Zhu J, et al. MobileOne: An Improved One Millisecond Mobile Backbone[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 7907-7917.
21. Stemmer U. Locally private k-means clustering[J]. The Journal of Machine Learning Research, 2021, 22(1): 7964-7993.
22. C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Work-shops (CVPRW), Jun. 2020, pp. 390–391.
23. K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, Jan. 2014.
24. S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8759–8768.
25. Chen J, Kao S, He H, et al. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 12021-12031.
26. Vadera S, Ameen S. Methods for pruning deep neural networks[J]. IEEE Access, 2022, 10: 63280-63300.
27. Gholami A, Kim S, Dong Z, et al. A survey of quantization methods for efficient neural network inference[M]//Low-Power Computer Vision. Chapman and Hall/CRC, 2022: 291-326.
28. Gou J, Yu B, Maybank S J, et al. Knowledge distillation: A survey[J]. International Journal of Computer Vision, 2021, 129: 1789-1819.
29. K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 1577–1586.
30. Yun S, Han D, Oh S J, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 6023-6032.
31. Zeng G, Yu W, Wang R, et al. Research on mosaic image data enhancement for overlapping ship targets[J]. arXiv preprint arXiv:2105.05090, 2021.
32. Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "SeaShips: A large-scale precisely annotated dataset for ship detection," IEEE Trans. Multimedia, vol. 20, no. 10, pp. 2593–2604, Oct. 2018.
33. L. Zhu, X. Geng, Z. Li, and C. Liu, "Improving YOLOv5 with attention mechanism for detecting boulders from planetary images," Remote Sens., vol. 13, no. 18, p. 3776, Sep. 2021.
34. Baltrušaitis T, Ahuja C, Morency L P. Multimodal machine learning: A survey and taxonomy[J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 41(2): 423-443.

Figures

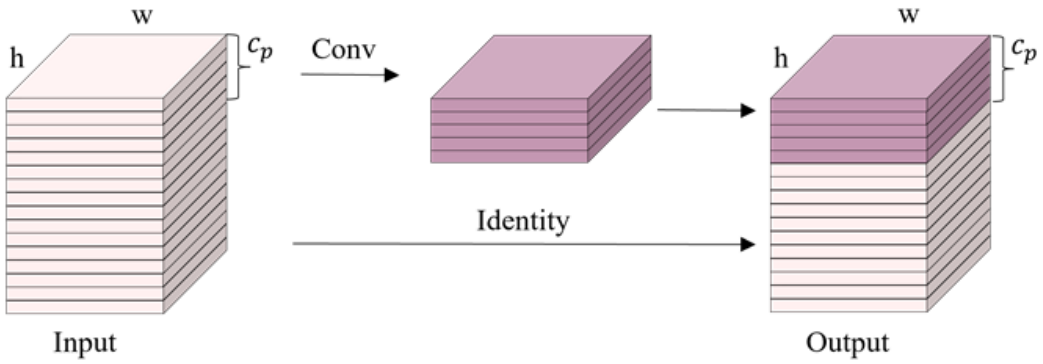

Figure 1

Module structure of PConv

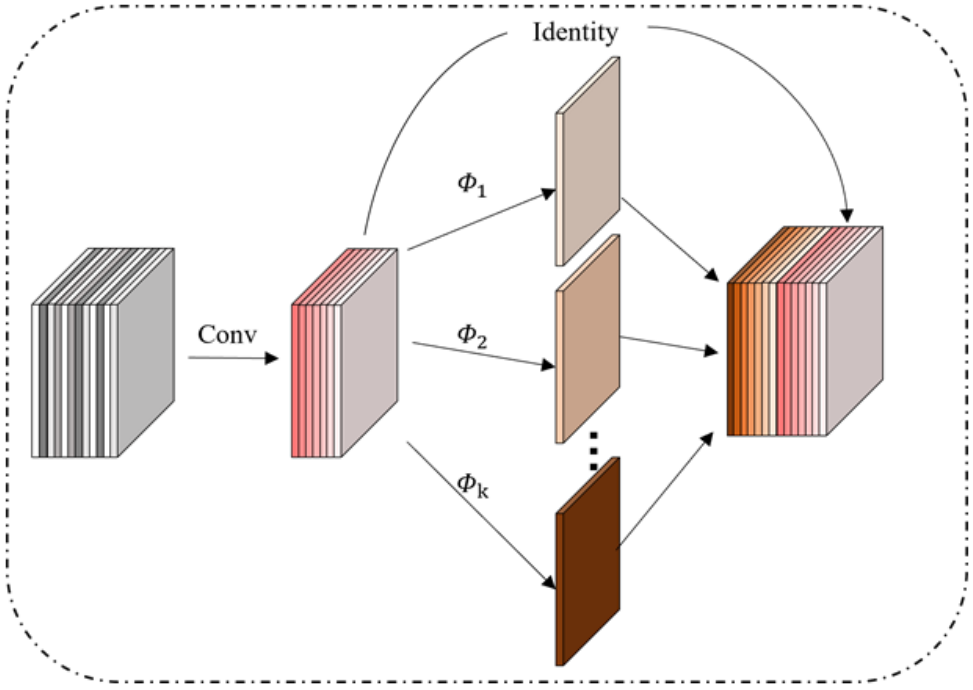

Figure 2

Module structure of GhostConv Φ denotes the cheap operations

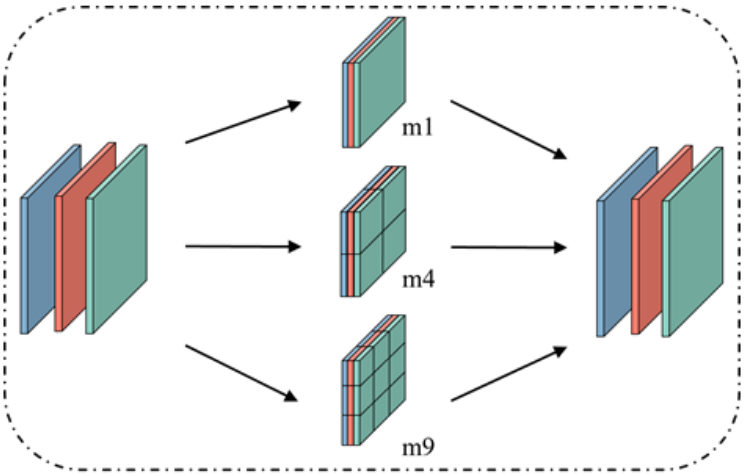

Figure 3

Module structure of Mosaic-9

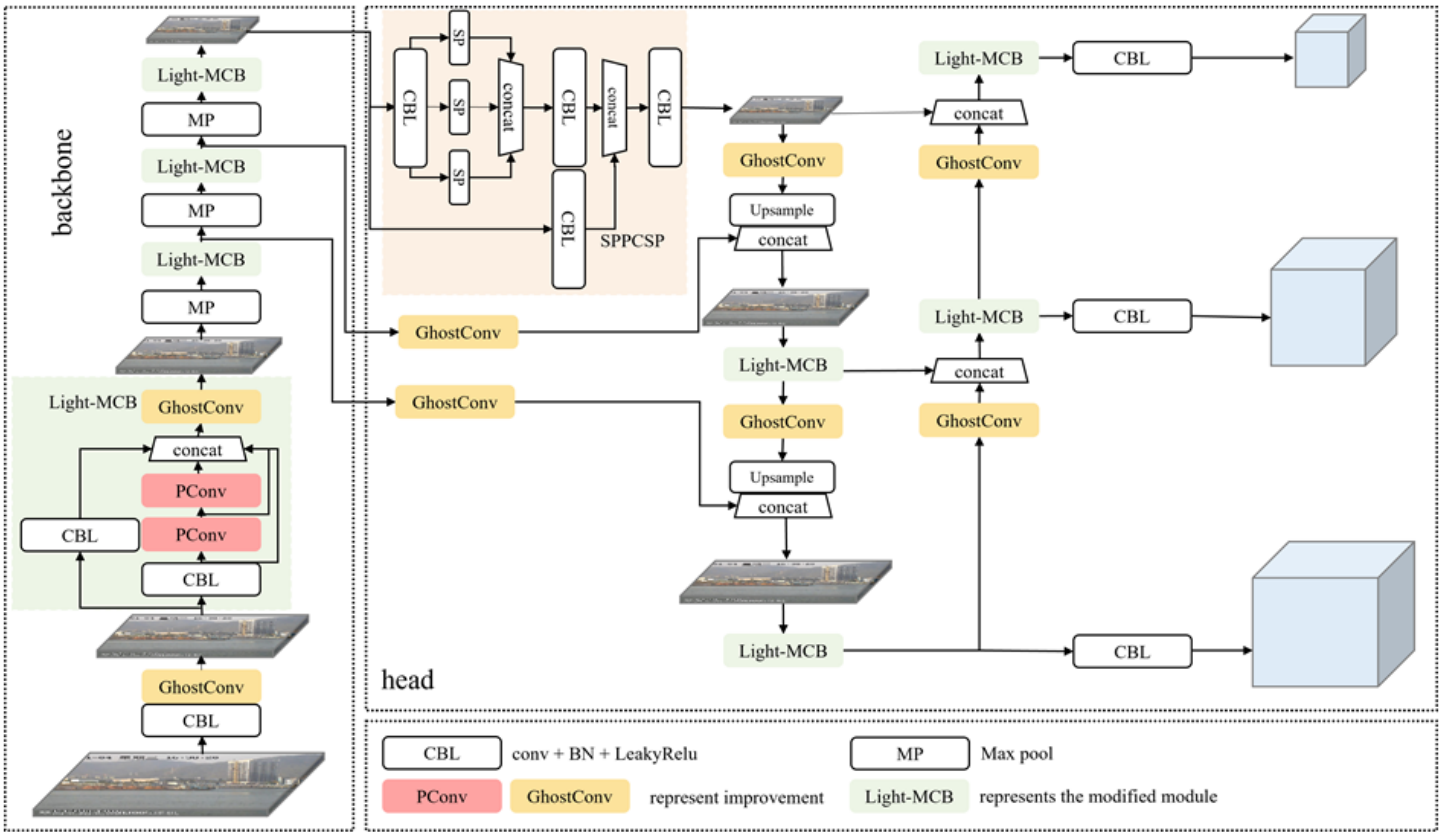

Figure 4

Module structure of Mosaic-9

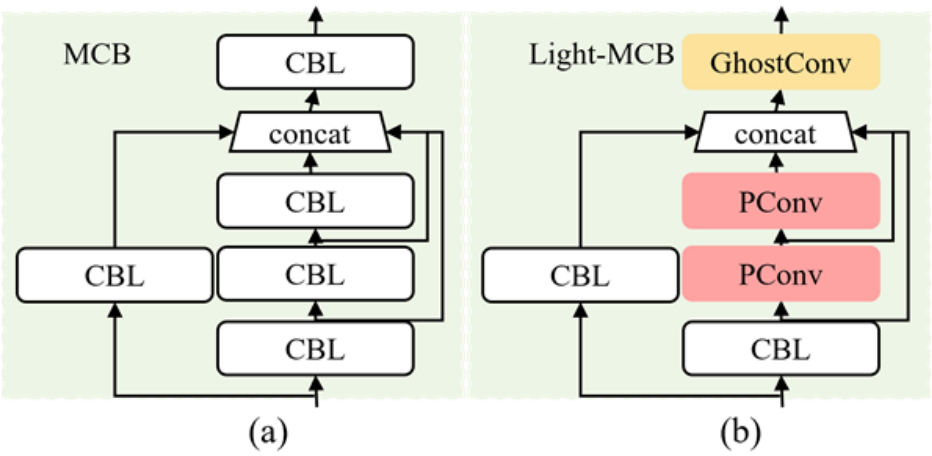

Figure 5

The structure comparison between MCB and proposed light-MCB. (a) MCB; (b) proposed Light-MCB

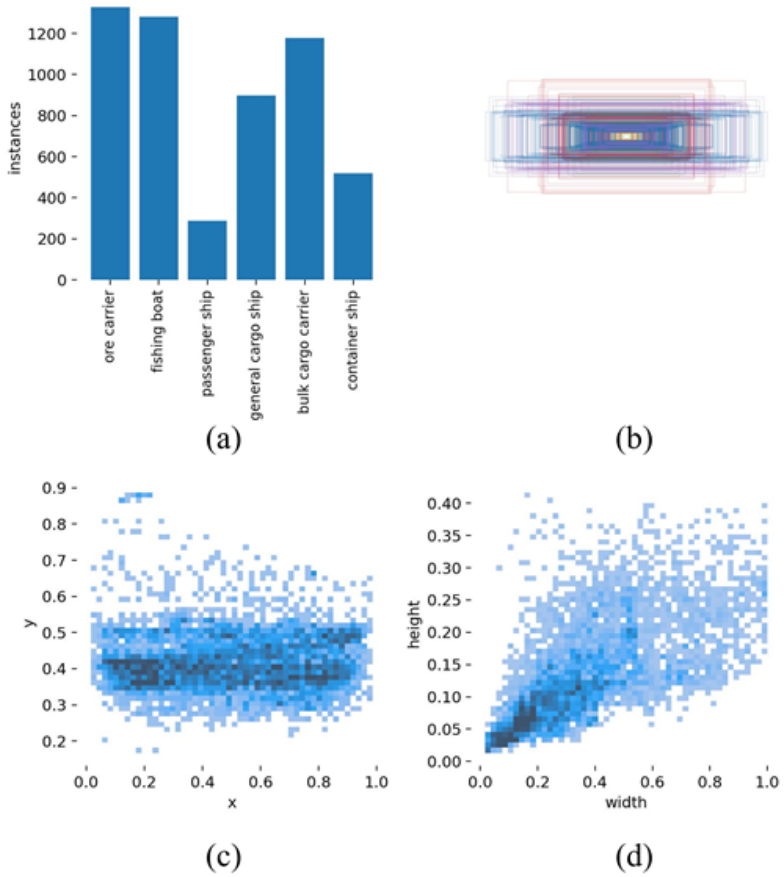

Figure 6

The distribution of SeaShips7000. (a) The number of six types of ship instances; (b) The visualization of the candidate boxes; (c) The center coordinates of the instances; (d) The height and width of ship instances

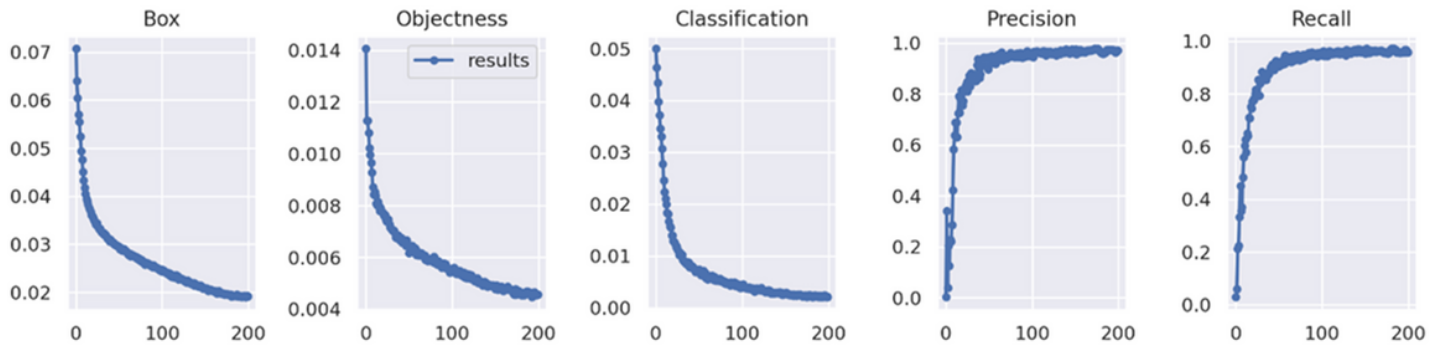

Figure 7

Loss curve of the LSDNet

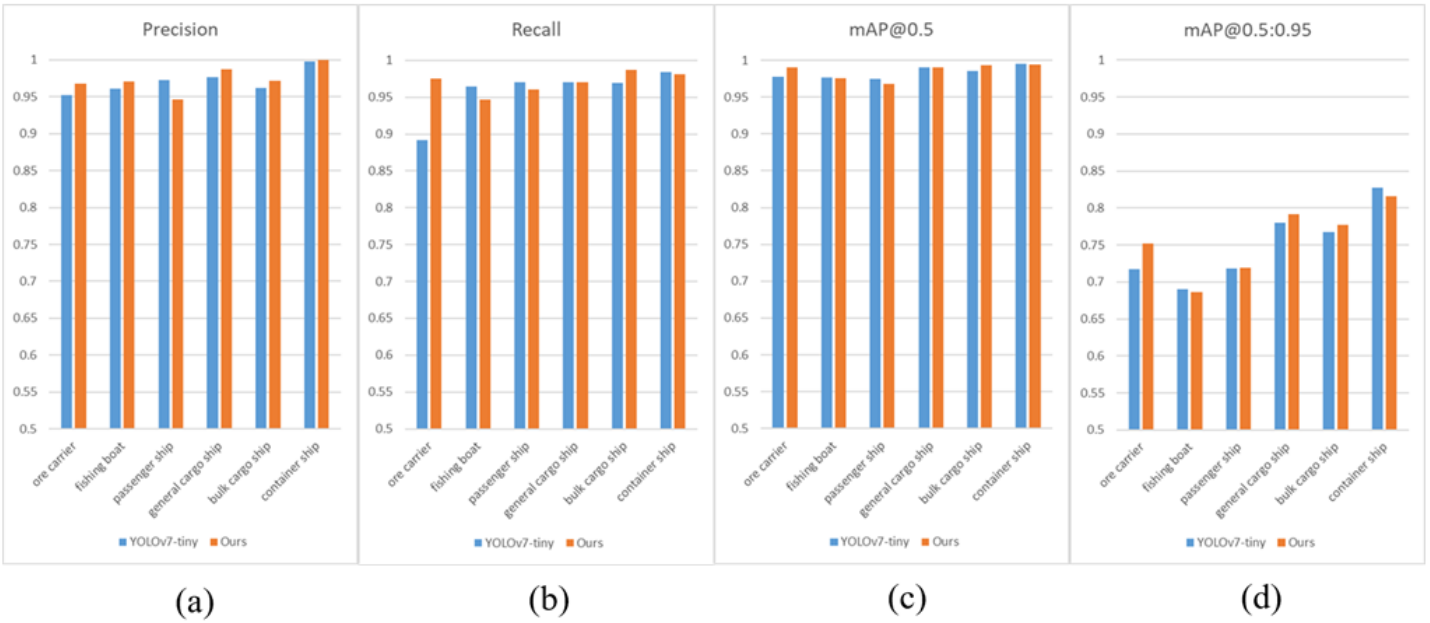

Figure 8

Detection results of different kinds of ship. (a) Precision; (b) Recall; (c) mAP@0.5; (d) mAP@0.5:0.95

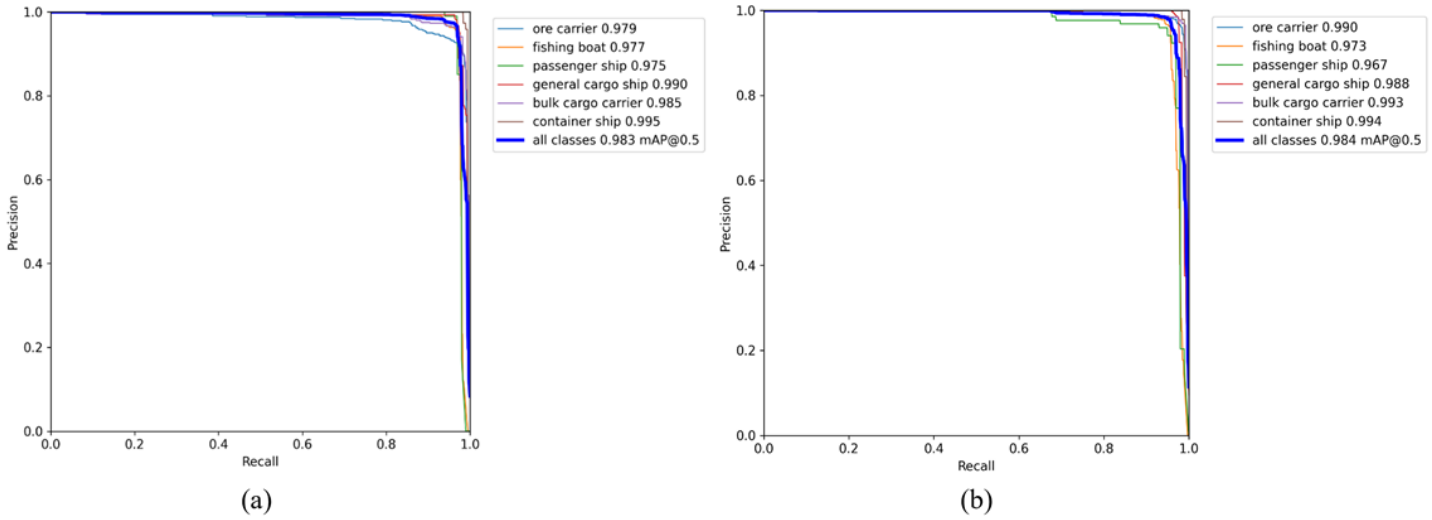

Figure 9

P-R curve of different kinds of ship. (a) Original; (b) LSDNet

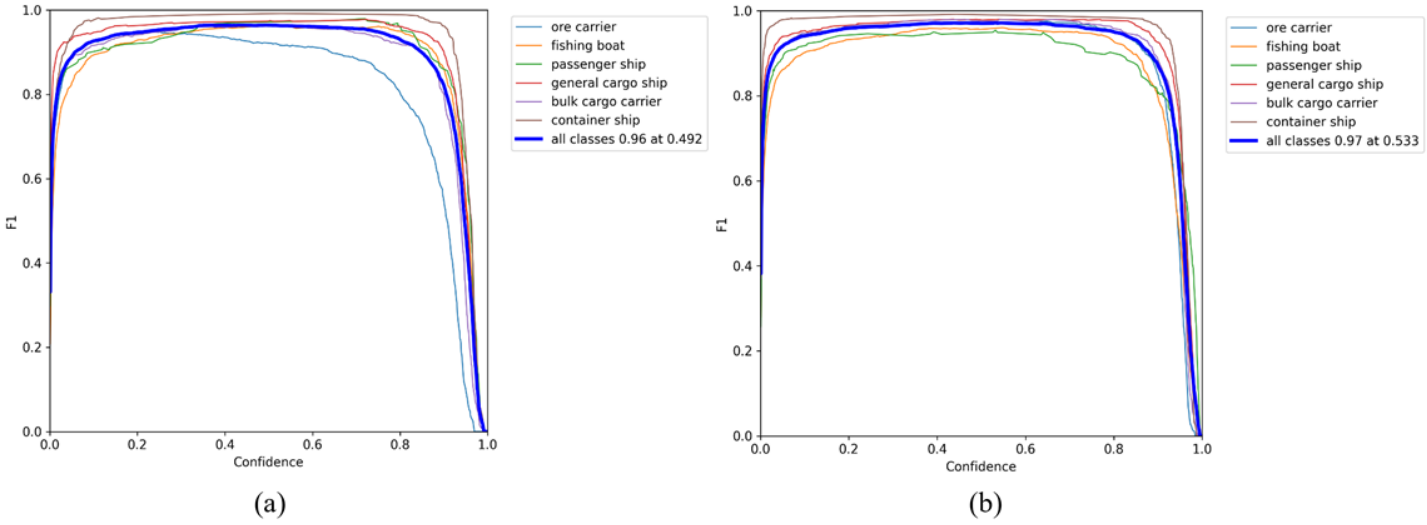

Figure 10

F1-score curve of different kinds of ship. (a) Original; (b) LSDNet

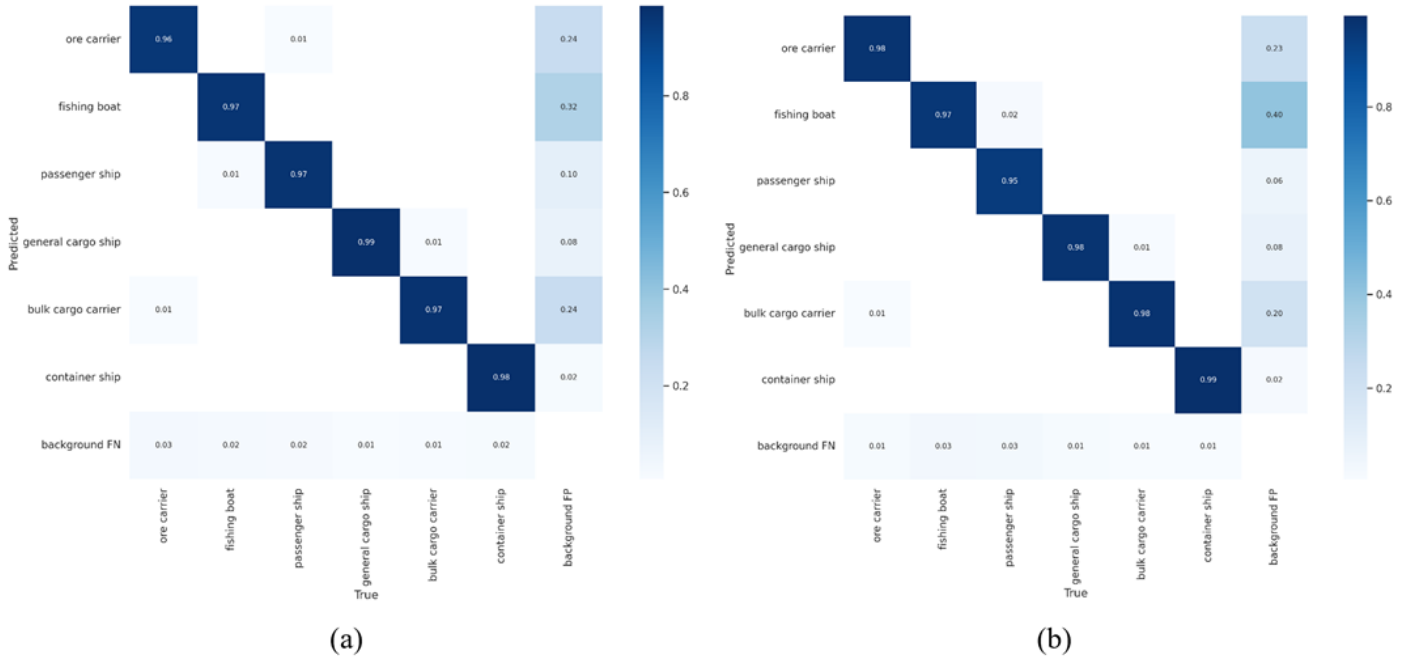

Figure 11

Confusion matrix for the validation set with various ships. (a) Original; (b) LSDNet

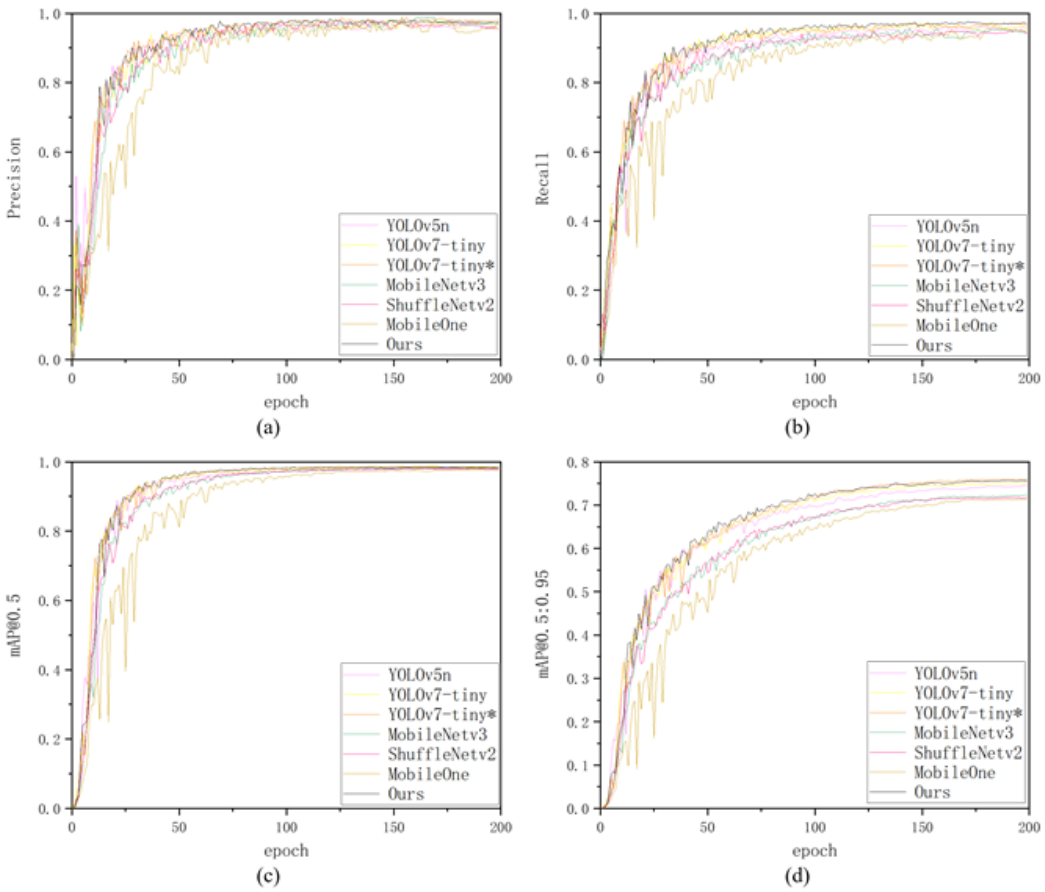

Figure 12

Comparison of Our Net with other models on the SeaShips7000 dataset. (a) Precision; (b) Recall; (c) mAP@0.5; (d) mAP@0.5:0.95

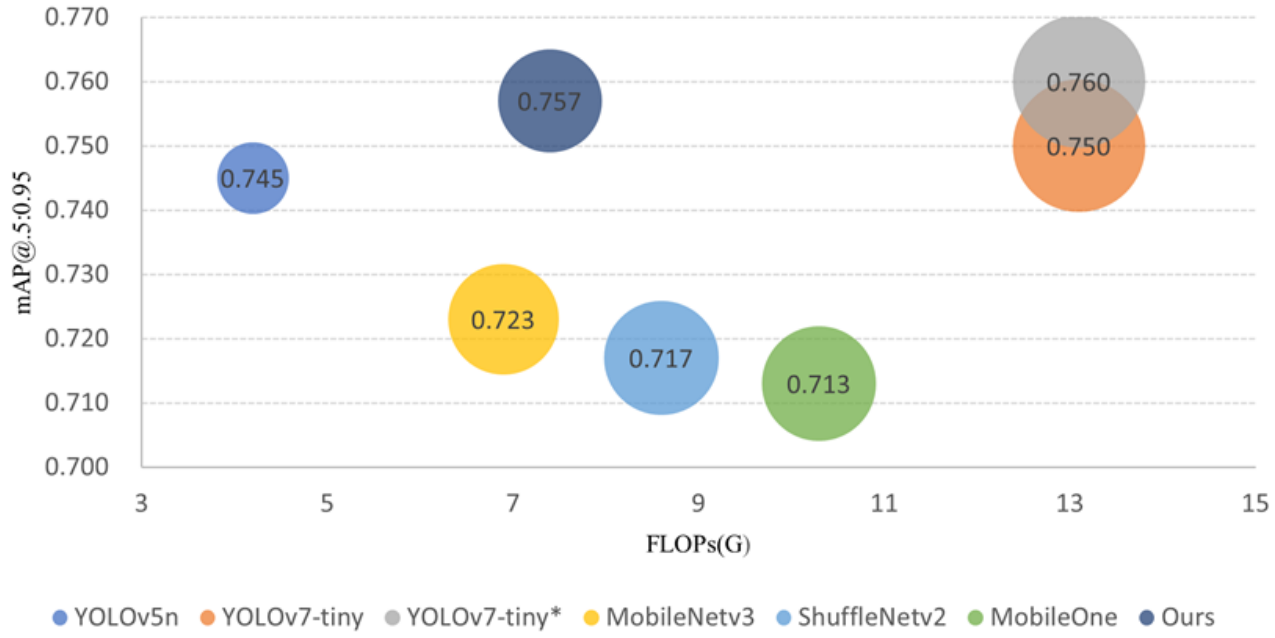

Figure 13

Comparison of the proposed LSDNet with other models. Each bubble's area represents the total number of parameters. MobileNetv3/ShuffleNetv2/MobileOne replaces the Backbone of YOLOv7-tiny, respectively

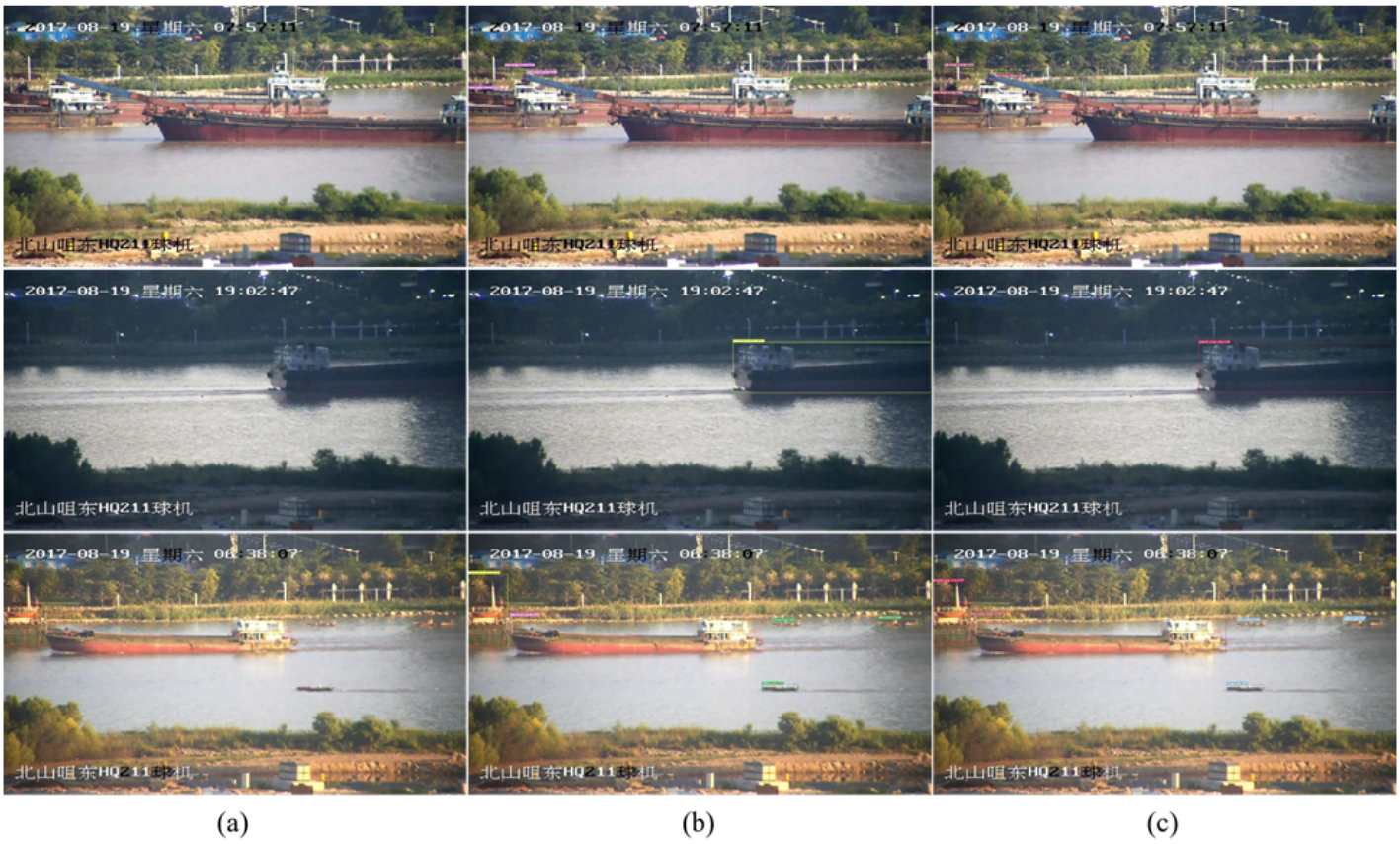

Figure 14

Visual comparison of Our Net with original YOLOv7-tiny. (a) Original; (b) YOLOv7-tiny; (c)Ours