remote sensing

Article

Lightweight Underwater Object Detection Based on YOLO v4 and Multi-Scale Attentional Feature Fusion

Minghua Zhang [1], Shubo Xu [1], Wei Song [1,*], Qi He [1] and Quanmiao Wei [2]

[1] College of Information Technology, Shanghai Ocean University, Shanghai 201306, China; mhzhang@shou.edu.cn (M.Z.); m200711475@st.shou.edu.cn (S.X.); qihe@shou.edu.cn (Q.H.)
[2] East China Sea Bureau, Ministry of Natural Resources, Shanghai 200137, China; qmwei@eastsea.gov.cn
* Correspondence: wsong@shou.edu.cn

Abstract: A challenging and attractive task in computer vision is underwater object detection. Although object detection techniques have achieved good performance in general datasets, problems of low visibility and color bias in the complex underwater environment have led to generally poor image quality; besides this, problems with small targets and target aggregation have led to less extractable information, which makes it difficult to achieve satisfactory results. In past research of underwater object detection based on deep learning, most studies have mainly focused on improving detection accuracy by using large networks; the problem of marine underwater lightweight object detection has rarely gotten attention, which has resulted in a large model size and slow detection speed; as such the application of object detection technologies under marine environments needs better real-time and lightweight performance. In view of this, a lightweight underwater object detection method based on the MobileNet v2, You Only Look Once (YOLO) v4 algorithm and attentional feature fusion has been proposed to address this problem, to produce a harmonious balance between accuracy and speediness for target detection in marine environments. In our work, a combination of MobileNet v2 and depth-wise separable convolution is proposed to reduce the number of model parameters and the size of the model. The Modified Attentional Feature Fusion (AFFM) module aims to better fuse semantic and scale-inconsistent features and to improve accuracy. Experiments indicate that the proposed method obtained a mean average precision (mAP) of 81.67% and 92.65% on the PASCAL VOC dataset and the brackish dataset, respectively, and reached a processing speed of 44.22 frame per second (FPS) on the brackish dataset. Moreover, the number of model parameters and the model size were compressed to 16.76% and 19.53% of YOLO v4, respectively, which achieved a good tradeoff between time and accuracy for underwater object detection.

Keywords: YOLO; lightweight network; underwater object detection; attention mechanism

Citation: Zhang, M.; Xu, S.; Song, W.; He, Q.; Wei, Q. Lightweight Underwater Object Detection Based on YOLO v4 and Multi-Scale Attentional Feature Fusion. *Remote Sens.* **2021**, *13*, 4706. https://doi.org/10.3390/rs13224706

Academic Editor: Jaroslaw Tegowski

Received: 16 September 2021
Accepted: 15 November 2021
Published: 21 November 2021

1. Introduction

The marine environment is a complicated system. As the underwater environment is very different from the land environment, some techniques of remote sensing including acoustic, magnetism [1], and 3D shallow seismic [2] sensing have achieved good performance in the marine realm. With the development of computer vision, exploring the oceans with computer vision technology has become a new avenue. As a basic task of computer vision, object detection based on optical imaging has become an absorbing work in the marine realm. In recent years, many researchers have begun to study underwater object detection based on optical imaging and have achieved great results. At present, underwater object detection has many applications in the marine environment, including the study of marine ecosystems, marine biological population estimation, marine species conservation, pelagic fishery, underwater unexploded ordnance detection [3], underwater archaeology and many other potential applications, providing an effective way to exploit marine resources.

Although object detectors based on deep convolutional neural networks (DCNN) have performed well on general category datasets in recent years, it is not efficient to apply them directly to underwater scenarios due to the slow speed and large model size of large networks. As the underwater scenery is more complex than the land scene, the image samples obtained through underwater camera equipment are small and generally of low quality. On the one hand, these images often suffer from high noise, low visibility, blurred edges, low contrast and color cast. Moreover, underwater targets are usually clustered and small, which brings additional challenges to studies involving underwater object detection. On the other hand, underwater embedded devices have limited storage and computing power, which makes large detection networks ineffective in underwater environments. However, there is little research on lightweight underwater object detection. The approach of focusing on improving the detection accuracy of underwater targets through large networks could be revisited. Therefore, it is necessary to study a lightweight detector that has a high detection accuracy and small model size for underwater object detection.

The detection of underwater small targets is a tough problem. Improving the detection accuracy of small targets will make a great contribution to overall detection performance. As it turns out, the feature information of small targets will gradually decrease or even disappear in the process of down-sampling with the increasing of the network layers. Many researchers have tried to extract more shallow features [4] or propose effective feature fusion methods [5,6] to better detect targets of different scales. However, most efforts have focused on constructing complex paths to combine features from different convolutional kernels, groups, and layers. Current multi-scale feature fusion methods are usually implemented by adding or concatenating features from different layers; they only provide fixed linear aggregation of feature mapping, and they are context-independent. Dai, Y et al. [7] proposed a multi-scale channel attention module (MS-CAM) to alleviate the problems caused by scale variations and small targets, which can simultaneously emphasize both large targets with more global distributions and small targets with more local distributions. This module makes it better to identify targets on extreme scales. However, this module cannot change the number of features, which is limited in embedding different feature fusion structures. Therefore, the modified attentional feature fusion (AFFM) module based on the attentional feature fusion (AFF) [7] module is designed by adopting a 1×1 convolution to expand the feature channels after fusing feature information instead of using the adding method in AFF, which aims to not only enable cross-channel information interactions but also to easily define the number of output features compared to the AFF module. This module can be easily applied to different structures. Experiments indicate that it can effectively improve accuracy after fusing multi-scale feature information.

The existing object detection algorithms can be divided into two categories: one-stage [4,8] and two-stage [9–11] algorithms. The former is faster but has a lower accuracy, while the latter is more accurate but slower. In order to ensure a sufficient real-time detection performance, the framework of the YOLO v4 algorithm [12] has been adopted in our work. Although the YOLO v4 algorithm has achieved excellent performance by using the stronger base of CSPDarknet53 [13] with several amendments, it has the disadvantages of a huge number of parameters and a large model size. It is not practical enough to directly apply the detectors based on YOLO v4 in marine underwater environments. In order to address the problem of building a lightweight underwater object detector while maintaining a relatively high accuracy, a combination of a lightweight backbone, MobileNet v2 [14], and a depth-wise separable convolution [15] has been proposed to significantly decrease the amount of network parameters. Meanwhile, the attention mechanism has been introduced in our work to improve the feature pyramid network (FPN) structure [5] used in YOLO v4, which has achieved a higher accuracy. Experimentally, the proposed method achieved a mean average precision (mAP) of 81.67% and 92.65% on the PASCAL VOC dataset [16] and the brackish dataset [17], respectively, and the detection speed exceeded 44 frame per second (FPS), with the number of model parameters and model size compressed to 16.76% and 19.53% of YOLO v4, respectively.

A few key points of our work are listed as follows.

(1) Fewer parameters: Rebuilding of the network structure of YOLO v4 by combining MobileNet v2 with depth-wise separable convolution, which obtained a faster detection speed and greatly reduced the number of model parameters.

(2) Better feature fusion: AFFM was designed for the FPN structure to fuse and enhance features at different scales with a small cost, which was beneficial to the detection of underwater small targets.

(3) Multi-scale lightweight detector: Building of a multi-scale lightweight marine underwater object detector that showed a great tradeoff between accuracy and speed.

2. Related Works

2.1. Object Detection

At present, although object detection has been successful for generic category datasets, it is still an arduous challenge for the task of underwater object detection. In underwater scenes, the quality of underwater images is greatly affected by illumination, which results in low visibility, low contrast and color distortion. Meanwhile, the complex underwater environment and the small underwater objects make the detection of underwater targets more difficult.

Due to the differences between underwater and land-based optical imaging principles, underwater object detection tasks are generally composed of image pre-processing and object detection. Image pre-processing, i.e., image enhancement and restoration according to the specific task, aims to improve image quality for the latter process. The object detection framework includes four parts including: input, backbone, neck and head. The input is where the input image is defogged, denoised, enhanced and normalized according to the specific situation in order to enhance the features of the target of interest and to weaken background features. The backbone is a kind of network that extracts features from objects of interest in the input images, such as the classical AlexNet, VGGNet, GoogleNet and the recent hotspots of ResNet [18] and DenseNet [19], as well as SqueezeNet [20], MobileNet (V1, V2) [14,15], and ShuffleNet [21], etc., which are designed for light-weighting. The necks are usually situated between the backbone networks and the output layers for specific tasks, such as spatial pyramid pooling (SPP) [22], FPN [5], path aggregation networks (PANets) [6], etc., which enhance feature maps that contain both rich semantic information and deterministic location information. The head is the part that decodes and predicts the feature maps obtained from the above networks at multiple scales, which can be generally classified into anchor-based and anchor-free mechanisms in the two major representative algorithms: one-stage and two-stage algorithms.

In recent years, several outstanding achievements have been made by numerous researchers in fields related to underwater object detection. In the study of underwater image pre-processing, Wei-Hong Lin [23] et al. advanced an image enhancement algorithm based on candidate frame fusion for underwater target detection. Pritish Uplavikar [24] et al. put forward an underwater image enhancement algorithm using domain adversarial learning, which provides underwater target detection algorithms with better learning data. In research on underwater object detection algorithms, Xin Sun [25] et al. used transfer learning to identify objects in low-quality underwater videos, which achieved an average classification accuracy of 99.68% for 23 fish species and a video detection speed of 23 FPS. Fengqiang Xu [26] and Tien-Szu Pan et al. [27] conducted research on the multi-scale problem of underwater object detection by combining high-level semantic features with low-level spatial features to detect objects at different scales; the former achieved 79.13% mAP and 4 FPS on the PASCAL VOC, while the latter achieved 92.3% mAP and 23 FPS on the Fish4Knowledge dataset [28]. Long Chen et al. proposed SWIPENet [29], a network for underwater small sample detection. They used a sample re-weighting algorithm to reduce the weight of lost targets, which resulted in reductions in the interference of these noisy samples; Kai Hu et al. [30] put forward a sea urchin detection algorithm based on the SSD algorithm for feature enhancement.

As can be seen, underwater object detection is a very challenging task, but most present studies of marine underwater object detection aim at improving detection accuracy or image quality in complex marine environments. However, lightweight underwater object detection has attracted little attention even though lightweight object detection is successful in other fields. Lightweighting of underwater object detection is an interesting direction because of its practical engineering value. Therefore, how to perform accurate, fast and stable detection using lightweight framework in a complex underwater environment is an issue well worth investigating.

2.2. Lightweight Networks

In recent years, DCNN have shown broad prospects for deep learning and have achieved higher classification accuracies, and have been adopted in many tasks of computer vision. However, the storage and computation of neural network models on mobile and embedded devices remains a huge challenge due to the limitations of storage space and computing power. Lightweight networks, which aim to reduce the number of model parameters and computation complexity while maintaining accuracy, have become a hotspot in computer vision.

Current research on lightweighting of neural network models is divided into two mainstream directions: designing lightweight networks and model compression. The former, such as SqueezeNet, MobileNet (V1, V2) [14,15], ShuffleNet [21], Xception [31], etc., aims to design different convolutional methods and structures to make convolution neural networks more lightweight. The latter compresses models by means of knowledge distillation and pruning, etc. Geoffrey Hinton [32] et al. first introduced the concept of knowledge distillation, where they attempted to distill knowledge from multiple models to a single model in order to achieve knowledge migration and improve small model accuracy. Pruning of trained models is currently the most used method in model compression, which is usually used to find an effective criterion by which to judge the importance of parameters and to reduce the redundancy of models by cutting out unimportant channels or kernels.

Nowadays, lightweight networks are very popular in general object detection tasks. Take the MobileNet based SSD algorithms [15] for example, the general practice is to use a lighter backbone in a large detection network. Changing the convolution method is also a good way to reduce the number of model parameters [33,34].

Research into lightweight networks has a greater practical engineering value to some extent. However, there are few studies in the field of lightweight underwater object detection. Consequently, in this paper, we build a lightweight detector from the perspective of designing lightweight networks by combining MobileNet v2 with depth-wise separable convolution to better and more quickly detect underwater targets.

2.3. Multi-Scale Features Fusion for Small Object Detection

Early target detectors [8–11] tend to detect targets by utilizing features maps from the last fully connected layer of the convolutional neural networks (CNN). The higher layer features are usually more sensitive to large objects and have stronger semantics, but tend to miss small targets. What is more, the algorithms based on DCNN use many combinations of convolutions, followed by the pooling of layers in the hidden part of the networks, which reduces the original image resolution to a very small resolution. Due to this fact, the small target features they extract in the first layer, which are few to begin with, disappear somewhere in the network and never really reach the detection and classification steps, which makes it difficult to detect small targets. From this, Liu et al. [4] proposed to take advantage of features in different layers of the VGG16 [35] network to detect objects of different sizes, which makes it better at detecting small targets than YOLO v1 [8]. This approach proved that it was more effective than networks that use last-layer features for prediction in small target detection. It has been proven that shallow features are more sensitive to small targets, and that deeper features contain better semantic information. However, the semantic information of the shallow features extracted by SSD is not sufficient, and a structure that utilizes features from the forward position is needed. To make full use of both shallow and high-level features, Tsung-Yi Lin et al. [5] proposed an effective architecture, FPN, to fuse features from different layers. In contrast to the multiple feature branches of SSD, the FPN structure has two differences: a top-down layer-by-layer upsampling operation and a lateral connection. The FPN structure creatively takes advantage of both low-level and high-level features, which results in a feature map containing both good spatial information and strong semantics, and which achieves excellent performance. The way in which information is propagated through the neural network is important, but the FPN structure has a large span of information paths between low-level and high-level features. Building on this, PANet uses bottom-up path augmentation to shorten information paths and enhance multi-scale fusion information.

At present, these kinds of feature pyramid structures are widely used as a general component in object detection tasks. In our work, in order to alleviate the drawback of linear concatenation operations in FPN structures, the attention mechanism has been introduced into the FPN architecture for better detection of targets of different sizes.

2.4. Activation Functions

Activation functions aim to add some non-linear elements into the CNN to improve the expressive power of the model and to allow the neural network to better solve more complex problems.

The rectified linear unit (ReLU) and the ReLU6 are widely used in CNN due to the advantages of fast and simple calculations, which are formulated in Equations (1) and (2). However, ReLU and ReLU6 functions are directly truncated at negative values, and the gradient decline is not smooth enough. Diganta Misra et al. [36] proposed a new activation function, Mish, which does not completely truncate at negative values, but allows for a relatively small negative gradient inflow—thus ensuring information flow—and the Mish function also guarantees smoothing at every point, which makes the gradient descend more smoothly than in ReLU. The Mish function is formulated in Equation (3). A comparison of ReLU, ReLU6 and Mish functions is shown in Figure 1. It can be seen that the ReLU6 function avoids the problem of infinite growth shown by the ReLU function, but might result in saturation. Besides this, their derived functions are not smooth enough at every point, or are even zero in some intervals, which prevents gradient updating of negative values. This paper uses the Mish activation function to improve the ReLU6 activation function, which is very effective in improving its accuracy, although it increases the computational cost compared to ReLU6.

$$f_1(x) = max(\ 0,\ x)\ ,\ x \in R \tag{1}$$

$$f_2(x) = min(6, max(0, x))\ ,\ x \in R \tag{2}$$

$$f_3(x) = x * tanh(ln(1 + e^x))\ ,\ x \in R \tag{3}$$

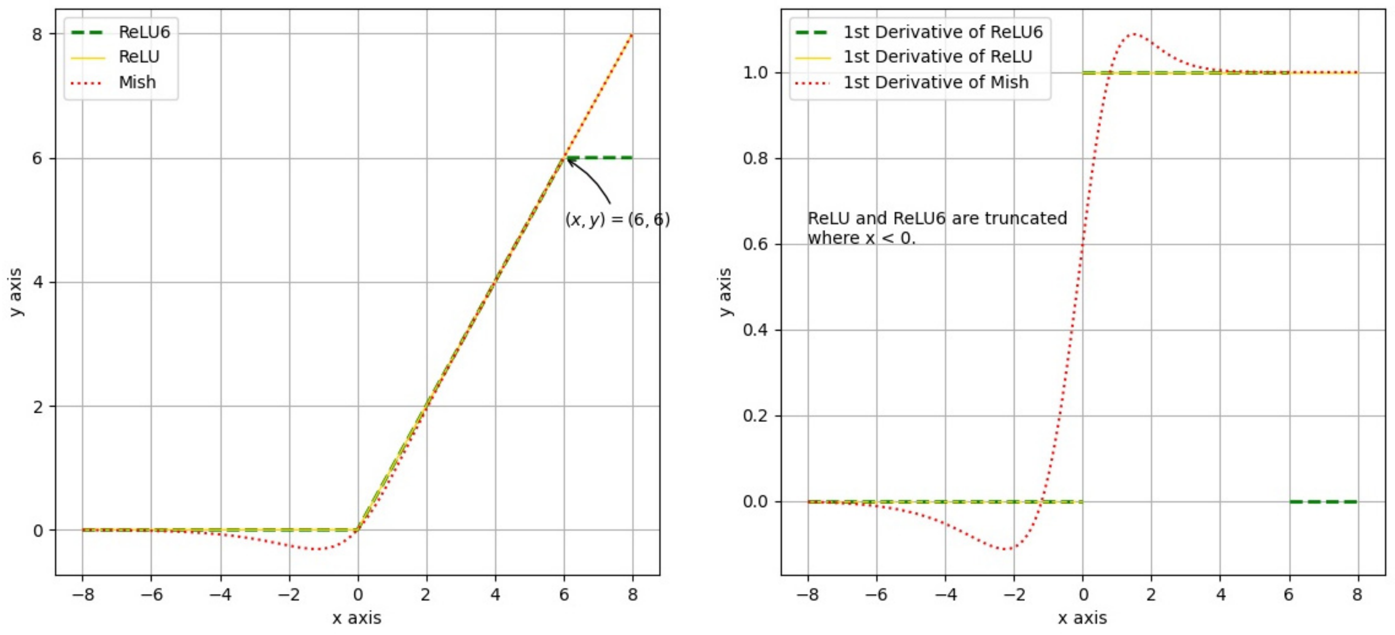

Figure 1. A comparison of ReLU, ReLU6 and Mish functions (**left**) and their 1st derivative function curves (**right**).

3. Methodology

This section details the main methods of the proposed lightweight detector. Section 3.1 describes the improved network structure as a whole. Sections 3.2 and 3.3 introduce the depth-wise separable convolution and the attentional feature fusion module, respectively.

3.1. Network Structure

The improved network structure can be divided into five parts: input, backbone, neck, head and output. Figure 2 shows the improved network structure. The main improvements of our work are in the parts of backbone, neck and head. In this paper, in order to achieve the purpose of lightweighting, MobileNet v2 was finally chosen as the backbone network because of its comprehensive performance after comparison to some other lightweight backbone networks, instead of the large CSPDarknet53 backbone network used in YOLO v4, and extracted features of three different scales to detect targets of different sizes. The feature map at a small scale had the highest semantic information and contained the weakest location information. The feature map at a large scale contained the richest location information and contained the weakest semantic information. The feature map at the middle scale contained medium location and semantic information in the three scales of extracted features. The SPP module was utilized to enhance the receptive field. The three preliminary feature maps were then enhanced by the FPN and the PANet structures for feature fusion.

In order to better fuse semantic and scale-inconsistent features, the AFFM was designed for the FPN structure in order to solve the problem of the linear adding and concatenation operations being context-independent and to enable cross-channel information interactions. To further decrease the number of model parameters, the neck and head structures were rebuilt with the depth-wise separable convolution, and finally the enhanced features of the three scales were fed into the YOLO detection head for prediction. Experiments show that the FPN structure increased the mAP by 1.08% on the PASCAL VOC dataset after using attention module.

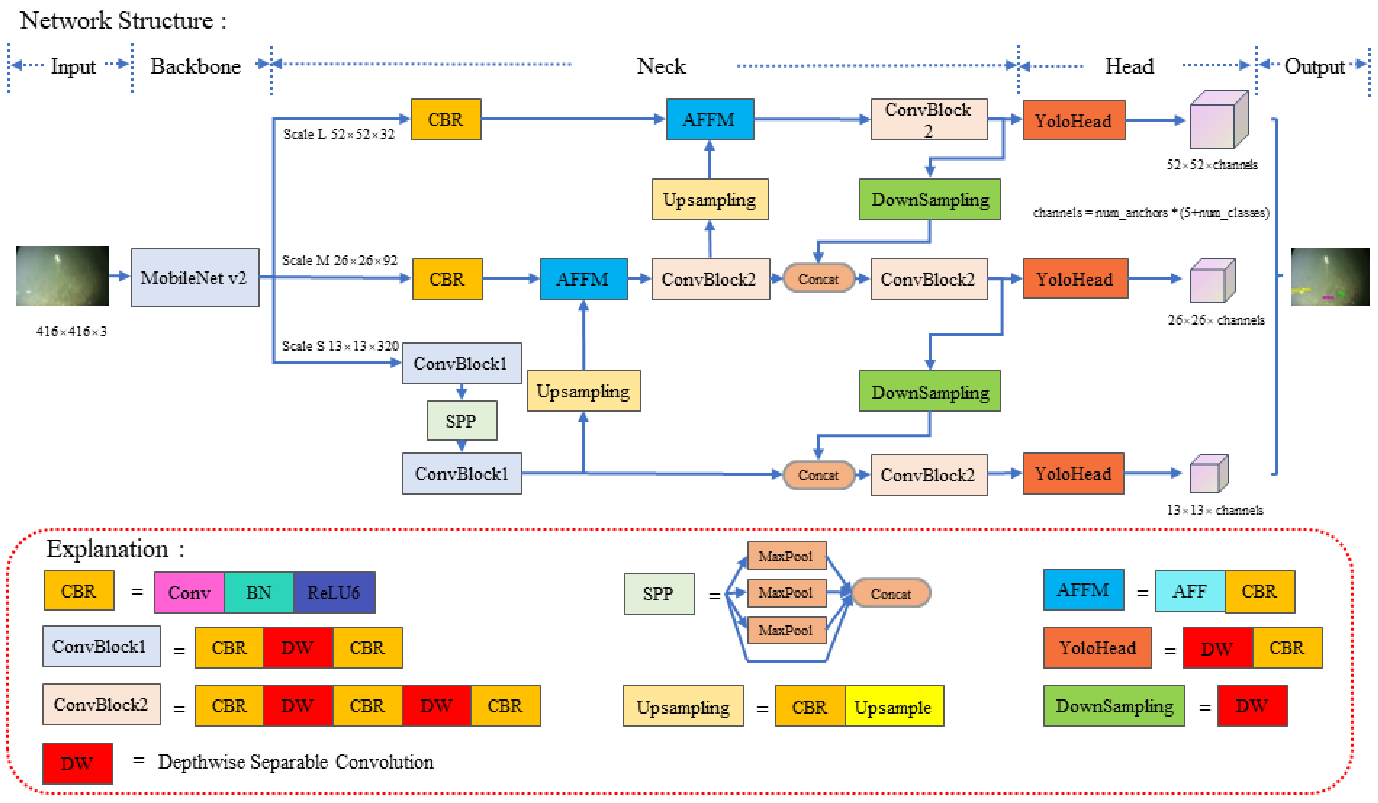

Figure 2. The improved network structure.

3.2. Depth-Wise Separable Convolution

Depth-wise separable convolution divides ordinary convolution processes into two processes of depth-wise convolution and point-wise convolution. The former first separates the channels to make the number of convolution kernels and input channels equal; with one convolution kernel only convolving one channel. The latter uses a 1 × 1 × M (M is the number of output channels in the previous layer) convolution kernel to blend the output feature map of the previous step, which compensates for the former's disadvantage of convolving each channel independently without using the information of different channels at the same spatial location. The use of depth-wise separable convolution significantly reduces the computational cost. The ratio of computational overhead to normal convolution was as shown in Formula (4):

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2} \tag{4}$$

$D_K \cdot D_K$ denotes the convolution kernel size. M and N represent the input and output channels, respectively. $D_F \cdot D_F$ is the size of feature map. It can be seen that the parameters and computational cost of the network were significantly reduced.

Figure 3 gives the specific convolution process for ordinary convolution and depth-wise separable convolution when the input shape is 5 × 5 × 3 and the convolution kernel size is 3 × 3. For ordinary convolution, the N convolution kernels of size 3 × 3 × 3 are moved 3 × 3 times, so the computation of ordinary convolution is N × 3 × 3 × 3 × 3 × 3 = 243N and the number of parameters is N × 3 × 3 × 3 = 27N. For depth-wise separable convolution, the three 3 × 3 × 1 convolution kernels are moved 3 × 3 times during the depth-wise convolution process and the number of multiplications is 3 × 3 × 3 × 1 × 3 × 3; In point-wise convolution, N convolution kernels of size 1 × 1 × 3 are moved 3 × 3 times, and the number of multiplications is N × 1 × 1 × 3 × 3 × 3 × 3. Therefore, the computation cost of depth-wise separable convolution is 3 × 3 × 3 × 1 × 3 × 3 + N × 1 × 1 × 3 × 3 × 3 = 243 + 27N, and the number of parameters is $3 \times 3 \times 3 \times 1 + N \times 1 \times 1 \times 3 = 27 + 3N$. Then, the ratio of computation cost and the number of parameters is $\frac{243+27N}{243N} = \frac{1}{N} + \frac{1}{9}$ and $\frac{27+3N}{27N} = \frac{1}{N} + \frac{1}{9}$, respectively. If $N = 128$ in this case, then the computational cost and the number of parameters of the depth-wise separable convolution is about 11.89% of the normal convolution, which indicates that the depth-wise separable convolution significantly reduced the computation cost and number of parameters. Figure 4 shows the convolution sequences of the depth-wise separable convolution and the normal convolution in a standard layer.

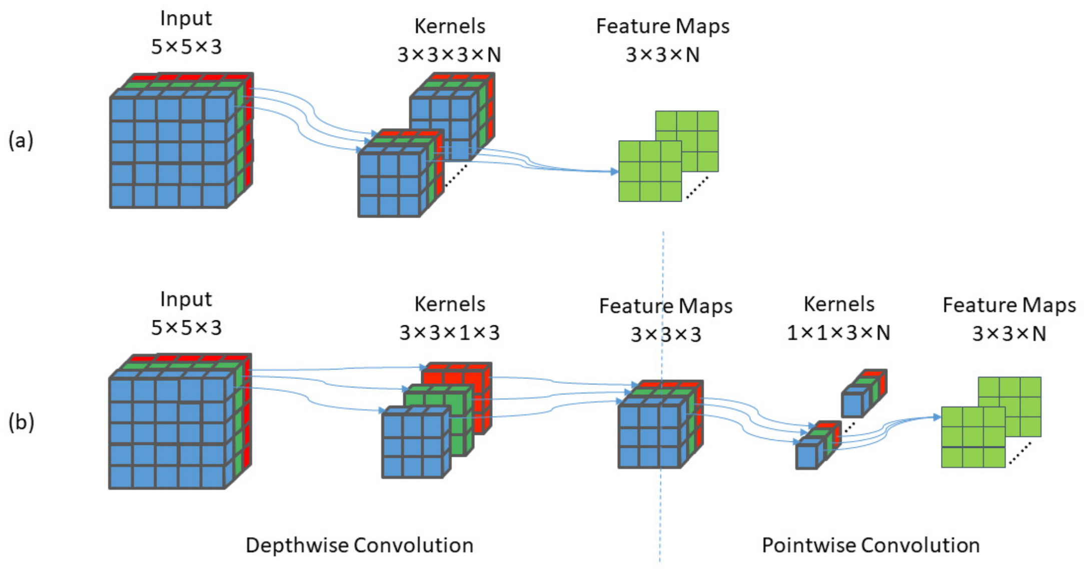

Figure 3. The specific process of convolution. (**a**): ordinary convolution, (**b**): depth-wise separable convolution.

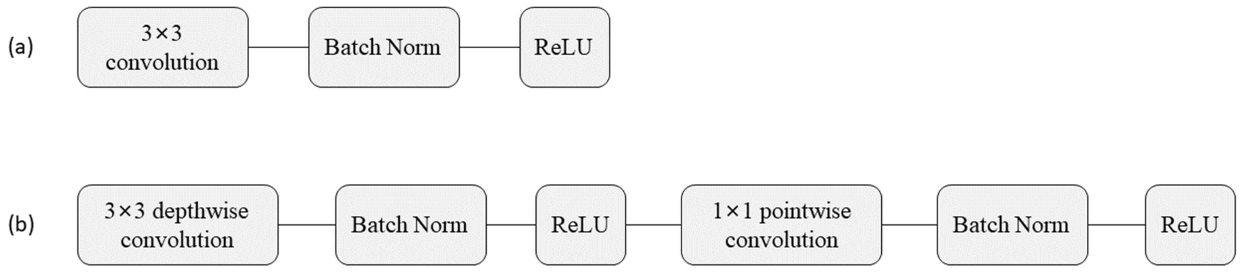

Figure 4. The convolution sequences in a standard layer. (**a**): normal convolution, (**b**): depth-wise separable convolution.

Rebuilding the neck and head parts with depth-wise separable convolution helped to decrease the number of parameters to 10.47m, which is about 27.03% of YOLO v4 with MobileNet v2 as the backbone, and 16.35% of original YOLO v4.

3.3. *Attentional Feature Fusion*

3.3.1. Multi-Scale Channel Attention Module

The MS-CAM makes use of two different branches to obtain channel attention weights. One branch uses global average pooling to investigate global features, while the other branch uses point-wise convolution to extract channel information for local features. Finally, the two branches are fused to better combine features at different scales. Figure 5 shows the structure of MS-CAM.

The refined features $Y \in R^{C\times H\times W}$ obtained by MS-CAM can be interpreted as Formula (5). $X \in R^{C\times H\times W}$ is the output of a layer in CNN. C represents the number of channels. $H \times W$ is the size of the feature map. $G(X)$ and $L(X)$ denote the global channel context and the local channel context, respectively. g(X) represents the global average pooling. The $G(X)$, $L(X)$ and g(X) can be explained as Formulas (6)–(8), respectively. $\mathcal{B}$ represents the batch normalization [37]. δ denotes the ReLU function, and σ represents the sigmoid function. The sizes of $PWConv_1$ and $PWConv_2$

are $\frac{C}{r} \times C \times 1 \times 1$ and $C \times \frac{C}{r} \times 1 \times 1$, respectively. $MSCAM(X) \in R^{C \times H \times W}$ represents the attention weights obtained by the MS-CAM module. $\oplus$ represents the addition operation of the broadcast mechanism and $\otimes$ indicates the element-wise multiplication operation.

$$Y = X \otimes MSCAM(X) = X \otimes \sigma(L(X) \oplus G(X)) \tag{5}$$

$$L(X) = \mathcal{B}(PWConv_2(\delta(\mathcal{B}(PWConv_1(X))))) \tag{6}$$

$$g(X) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X[:, i, j] \tag{7}$$

$$G(X) = \mathcal{B}(PWConv_2(\delta(\mathcal{B}(PWConv_1(g(X)))))) \tag{8}$$

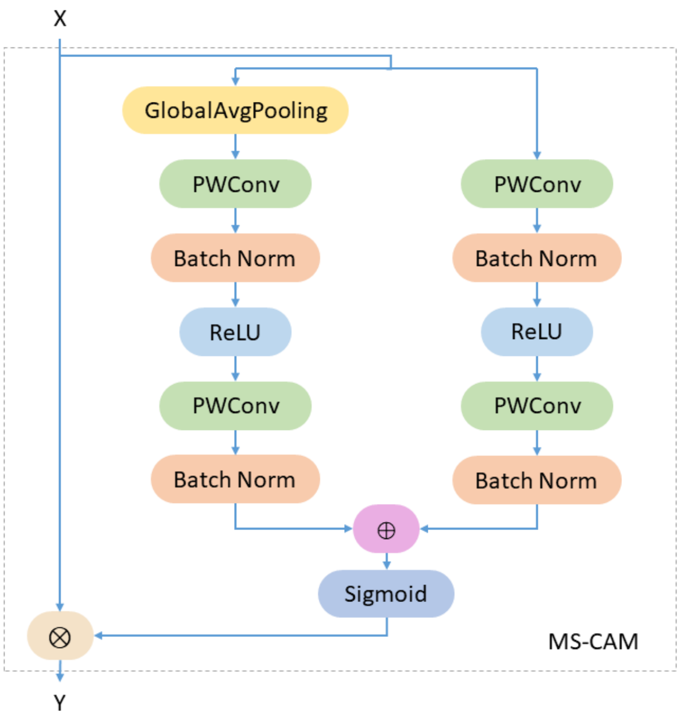

Figure 5. The main structure of MS-CAM [7].

3.3.2. Modified Attentional Feature Fusion Module

In order to better combine the MS-CAM module and to expand the feature channels, the AFFM is designed in this paper to better obtain contextual information from different convolutional layers by fusing semantic and scale-inconsistent features. Figure 6 shows the structure of the AFFM.

Given two features $X, Y \in R^{C \times H \times W}$, the AFFM can be represented as shown in Formula (9). The symbol $\uplus$ denotes the initial feature fusion using the operation of summing the corresponding elements. The sequence of 1×1 convolution, batch normalization and ReLU6 are denoted as CBR.

$$Z = CBR(MSCAM(X \uplus Y) \otimes X + (1 - MSCAM(X \uplus Y)) \otimes Y) \tag{9}$$

The redesigned AFFM aims to improve the FPN structure. Unlike the linear concatenation operation used in YOLO v4, the AFFM improved the generalization capability of the model by introducing non-linear operations for better fusion of semantic and scale-inconsistent features. The FPN with the AFFM (FPN-AFFM) structure is shown in Figure 7.

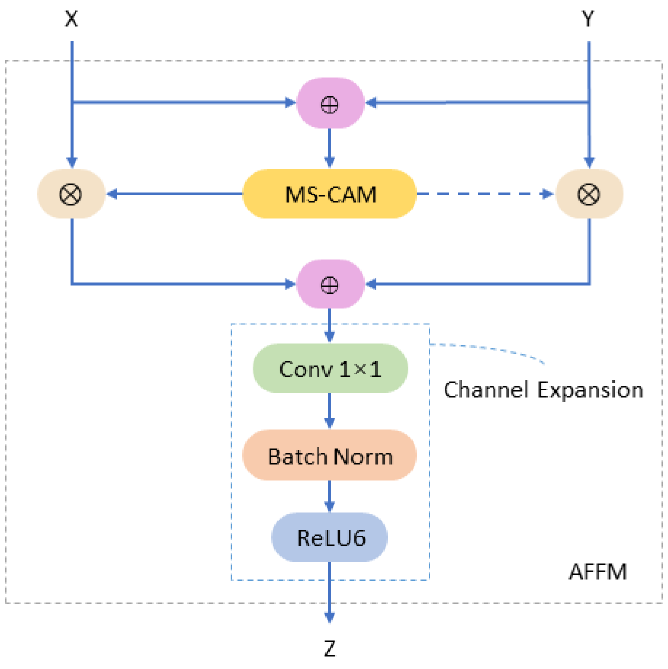

Figure 6. The main structure of the AFFM.

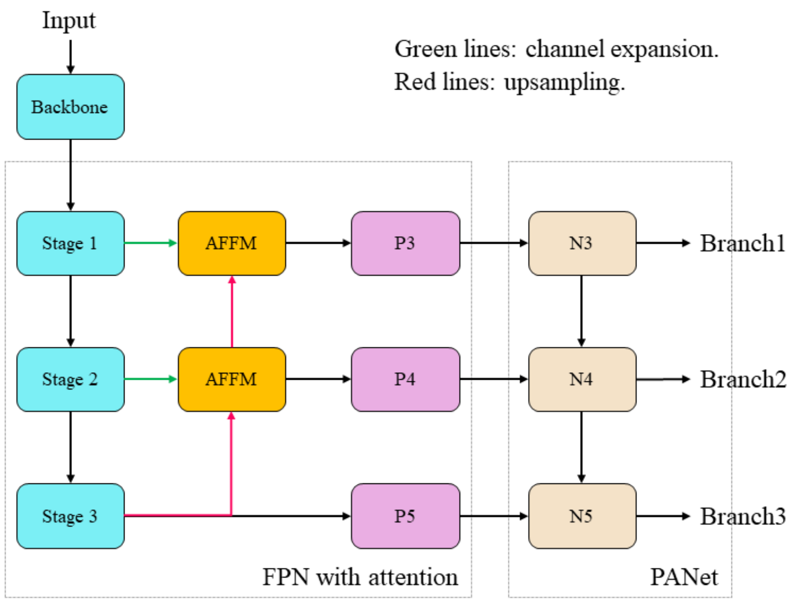

Figure 7. The main structure of FPN-AFFM.

The experiments show that the model has achieved a better detection performance by using the proposed AFFM, with an improvement of 0.78% mAP on the PASCAL VOC dataset compared with the YOLO v4 algorithm using MobileNet v2 and depth-wise separable convolution. At a very small cost, this method only has increased a small number of parameters and decreased by about 3.8 FPS in detection speed.

4. Experiments and Results

4.1. General Datasets and Underwater Image Datasets

4.1.1. PASCAL VOC Dataset

The PASCAL VOC dataset is a generic target detection dataset that has served as a standard dataset for measuring the performance of target detection algorithms over a long period of time. As this method is used for underwater scenes, the PASCAL VOC served as a pre-trained auxiliary dataset; it has VOC2007 and VOC2012, with a total of 20 common object classes.

In this paper, the transfer learning idea was adopted to pre-train the model on the PASCAL VOC2007 and VOC2012 datasets. The training-validation datasets of VOC2007 and VOC2012 were firstly mixed, containing 16,551 images in total, and then were divided into training and validation sets with a ratio of 9:1, and the test dataset of VOC2007 (4952 images in total) was used for testing. Finally, the pre-trained model was migrated to the underwater image dataset for later training.

4.1.2. Brackish Dataset

The brackish dataset [17] is an open-source underwater dataset that was created in 2019. This dataset is annotated based on real filmed underwater videos, which contain a total of six categories of underwater organisms: large fish, crabs, jellyfish, shrimps, small fish and starfish. The specific annotation information statistics are shown in Table 1. The visualization annotation of the images is shown in Figure 8.

Table 1. Category annotation number of the brackish dataset.

Species Category	Annotations	Video Occurrences
Big fish	3241	30
Crab	6538	29
Jellyfish	637	12
Shrimp	548	8
Small fish	9556	26
Starfish	5093	30

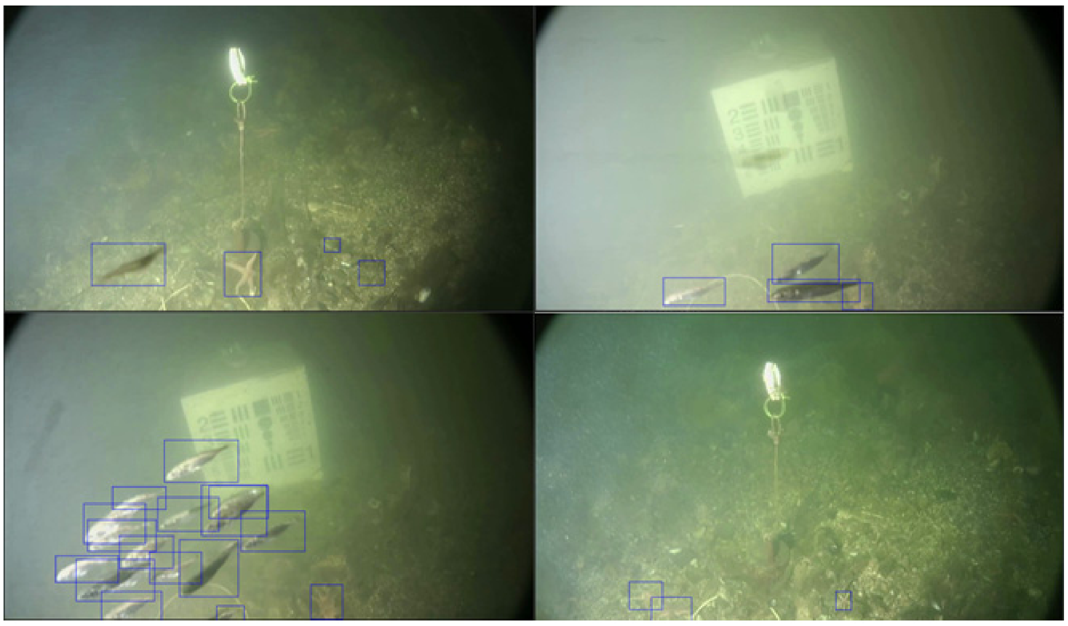

Figure 8. The visual annotations of the brackish dataset.

4.1.3. URPC Dataset

The original 2020 Underwater Robot Picking Contest (URPC) dataset includes five categories: echinus, holothurian, scallop, starfish and waterweeds, and contains 5543 training images. The waterweeds are annotated as a negligible category, which only contains 82 targets. Therefore, the waterweeds and unmarked images were removed. The final dataset used in this paper contained 5455 images and four categories, which were split up into a training set and a test set at a ratio of 9:1, because the test dataset of the URPC was not available. It is worth mentioning that the input resolution and category samples of this dataset are extremely unbalanced, which brings challenges in the training of the model. The annotation information statistics of the URPC 2020 dataset are shown in Table 2. Figure 9 shows the training images of the URPC 2020.

Table 2. Category annotation number of the URPC 2020 dataset.

Species Category	Annotations
Holothurian	5537
Echinus	22343
Starfish	6841
Scallop	6720

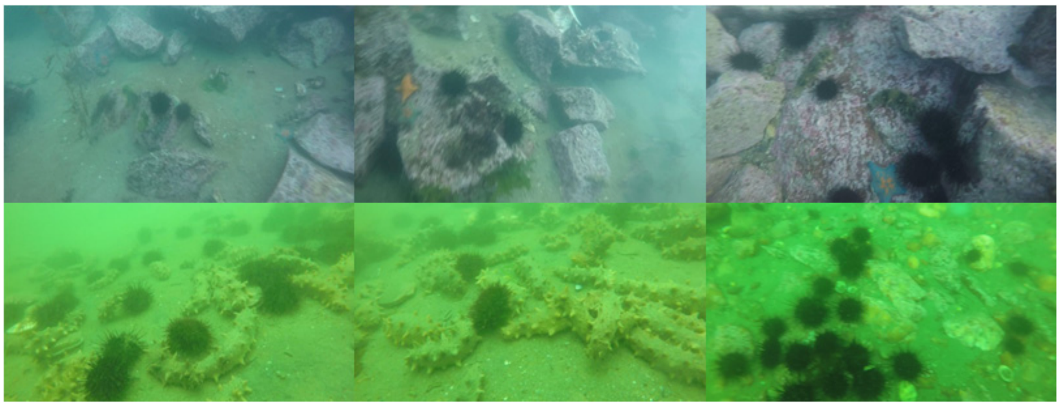

Figure 9. Training images from the URPC 2020.

4.2. Experimental Setup

4.2.1. Experimental Environment

The experimental environment is shown in Table 3. The hardware used for this experiment was an Intel(R) Xeon(R) Gold 6130 CPU, 2.10GHz; NVIDIA RTX 2080ti graphics card, single GPU, 11G memory; Ubuntu 16.04 operating system; CUDA version 10.2; PyTorch version 1.2.0. The compiled virtual environment of Python was v3.6.

Table 3. The environment for the experiments.

Environment	Versions or Model Number
CPU	Intel(R) Xeon(R) Gold 6130, 2.10 GHz
GPU	NVIDIA RTX 2080ti, Single GPU, Memory of 11G
OS	Ubuntu 16.04
CUDA	V 10.2
PyTorch	V 1.2.0
Python	V 3.6

4.2.2. Training Parameter Settings

In order to ensure fairness of the experiments, the same initial training parameters were set for each group of experiments. The input resolution was uniformly resized to 416×416. To prevent the initial backbone weights from being destroyed in the early stage of training, the method of freezing certain layers was used for training. In the process of freezing training, the gradients of the backbone network are not updated, and the training batchsize and epoch are set to 32 and 50, respectively. In the process of unfreezing training, the gradients of the whole network are all updated, and the batchsize and epoch are set to 16 and 150, respectively. The Adam algorithm [38] was adopted to optimize the loss function. The parameter settings in the Adam optimizer determined the training results. The weight decay coefficient was set to 5×10^{-4}. The initial learning rate was set to 1×10^{-3}, and the learning rate after unfreezing was set to 1×10^{-4}. Additional steps such as cosine annealing scheduling [39], mosaic data augmentation [12] and label smoothing were not used in our experiments, and were set to false. All other parameters were kept the same as in YOLO v4.

4.2.3. Testing Parameter Settings

The input resolution was uniformly resized to 416 × 416. The confidence threshold was set to 0.5 and the IOU threshold was set to 0.3. In the process of the speed test, just one GPU was used. The batchsize was set to 1, which means that only one image was processed at a time. The model inference time consisted of the processes of network inference, score screening and non-maximum suppression, and an average of 100 test times were taken as the final prediction time.

4.3. Experimental Results

4.3.1. Ablation Experiments

The model was trained and tested using the PASCAL VOC dataset. The mAP, a common index used in target detection, served as the evaluation criterion of accuracy. The evaluation criterion of model inference speed was FPS. Since this paper is about a study for model lightweighting, the number of network parameters and model size are also considered as the evaluation criterions.

To verify the effectiveness of the different submodules or means, ablation experiments were conducted in this paper. The main framework of the reimplemented YOLO v4 algorithm was used and the backbone network of YOLOv4 was replaced by MobileNet v2 at the same time. The rest network structures were consistent with YOLO v4.

Table 4 shows the results of the ablation experiments performed on MobileNet v2. The symbol (*, *) denotes the comparison of the gains based on baseline (the YOLO v4 that used the backbone of MobileNet v2.) and depth-wise separable convolution, and the symbol (*) denotes the gains compared with the baseline. In the listed models, Model1 (baseline) was the network replaced by the CSPDarknet53 backbone network with MobileNet v2. Model2 had rebuilding of the convolution structure of the neck and the head with the depth-wise separable convolution. Model3 was the model in which the proposed AFFM module was added to Model2. Model4 replaced the ReLU6 activation function of the CBR normal convolution module in Figure 2 with the Mish activation function.

The results indicated that the proposed Model2 respectively reduced the amount of model parameters and the model size by 72.97% and 69.72% compared with Model1, while the mAP and FPS were only decreased by 0.35% and 3.85, respectively. The proposed Model3 increased the mAP by 0.43% and 0.78% compared with Model1 and Model2, while only increasing 260k parameters, at 44.92 FPS. After replacing the activation function with Mish, the proposed Model4 achieved the best accuracy, i.e., the mAP reached 81.67%, while the detection speed was only reduced by 7.67 FPS compared to Model1 (baseline), and the mAP was improved by 0.94% and 1.29% compared to the baseline and Model2, respectively. As can be seen, the proposed method was effective, achieving a large gain at a small cost.

Table 4. Results of ablation experiments on the PASCAL VOC2007 test dataset.

Model	**Method**				**mAP(%) (*, *)**	**Parameters (M) (*)**	**Model Size (MB) (*)**	**Speed (FPS)**
	Baseline	**Dw**	**AFFM**	**Mish**				
Model1	√				80.73	38.74	154.6	51.85
Model2		√			80.38 (−0.35, 0)	10.47 (−72.97%)	46.8 (−69.72%)	48.00
Model3		√	√		81.16 (+0.43, +0.78)	10.73 (−72.30%)	47.8 (−69.08%)	44.92
Model4		√	√	√	81.67 (+0.94, +1.29)	10.73 (−72.30%)	47.8 (−69.08%)	44.18

Note: The symbol (*, *) denotes the gains compared with model1 and model2, respectively. The symbol (*) denotes the gains compared with model1.

4.3.2. Comparison with Other Object Detection Algorithms

A comparison with other object detection algorithms was conducted to verify the effectiveness of the proposed method. All results were tested on the VOC2007 test data. Table 5 shows the comparison results of our method and the other object detection algorithms on the PASCAL VOC dataset. The results suggest that the proposed method achieved a good tradeoff between precision and speed.

Table 6 shows the detailed detection results of 20 classes on the Pascal VOC 2007 test dataset between several detectors. Text in bold indicates the highest level. It is evident that our method achieved the highest AP among 11 classes.

4.3.3. Detection Results on Underwater Datasets

After pre-training, the obtained model Model4 was transferred to the underwater dataset for the final training, with total epoch set to 150 (freezing epoch set to 50 and non-freezing epoch set to 100), and other training parameters kept the same as those shown above. The training parameters for the YOLO v4 algorithm and the Tiny YOLO v4 algorithm were set the same to transfer the training. Table 7 shows the comparison results of our method with YOLO v4 and Tiny YOLO v4 on the brackish dataset. As can be seen, the number of model parameters and model size of the proposed method respectively increased by 4.77M and 25.1MB compared to the Tiny YOLO v4 algorithm in the brackish underwater dataset, and the increased number of parameters mainly focused on the FPN and the PANet structures. The FPS was 36.22% of the Tiny YOLO v4 algorithm, reaching 44.22, but mAP increased significantly by 12.49%, which significantly improved the accuracy of small targets. When compared to the YOLO v4 algorithm, the mAP was only reduced by 0.91%, but the number of parameters and model size were reduced by 83.2% and 80.5%, respectively, and the FPS was improved by 7.31. It is obvious that the proposed method achieved a balance between accuracy and speed, and made the detection model light enough.

In order to verify the applicability of the proposed method in different underwater environments, the parallel experiments were carried out on the URPC dataset. The results are shown in Table 8. It is obvious that our method achieved a 79.54% mAP, which was 1.47% lower than that of YOLO v4 and 11.71% higher than that of Tiny YOLO v4. The results of the experiments indicate that the proposed method can meet the demand of detecting targets in different underwater environments.

The detection results were visualized on the brackish and URPC underwater datasets in Figures 10 and 11. It was observed that the detection accuracy of our method was similar to the YOLO v4 algorithm, and the detection performance of small targets was significantly better than the Tiny YOLO v4 algorithm, while the Tiny YOLO v4 algorithm had a lot of missing detection cases in small target detection. The reason for this is that the Tiny YOLO v4 algorithm only extracted features at the scales of 13×13 and 26×26 and used an FPN structure for feature fusion to detect targets, so it was not sensitive enough to small targets.

To better explain the feature extraction ability of the proposed network, the feature maps of the network were also visualized. Figure 12 shows the original images of the brackish dataset. Figures 13 and 14 visualize the output feature maps at different scales of the first 12 channels for each kernel. Additionally, our models and YOLO v4 extracted 3 feature maps, and the model of Tiny YOLO v4 extracted only 2 feature maps. The results in Figures 13 and 14 prove that feature maps at small scales are better for large targets and that feature maps at large scales are sensitive to small targets. The feature maps of the proposed method have better texture and semantic information than Tiny YOLO v4, and have similar texture and semantic information to YOLO v4, which demonstrates the effectiveness of our method.

Table 5. The experimental results with different object detection algorithms.

Training Data	Method	Backbone	Input Size	mAP (%)	Parameters (M)	Model Size (MB)	GPU	Speed (FPS)
COCO [40] + 07 + 12	YOLO v4 [12]	CSPDarknet53	416 × 416	89.88	64.04	244.7	RTX 2080ti	36.14
	Tiny YOLO v4 [41]	Tiny CSPDarknet53	416 × 416	78.41	5.96	22.6	RTX 2080ti	123.51
07 + 12	Faster-RCNN [11]	VGG16	1000 × 600	73.2	134.70	~	K 40	7
	Faster-RCNN [18]	ResNet101	1000 × 600	76.4	~	~	Titan X	5
	SA-FPN [26]	ResNet50	1280 × 768	79.1	~	~	GTX 1080ti	4
	SSD300 [4]	VGG16	300 × 300	74.3	26.30	~	Titan X	46
	R-FCN [42]	ResNet50	1000 × 600	77.4	31.90	~	Titan X	11
	R-FCN3000 [43]	ResNet50	1000 × 600	79.5	~	~	P6000	30
	RON384++ [44]	VGG16	384 × 384	75.4	~	~	Titan X	15
	STDN321 [45]	DenseNet169	321 × 321	79.3	~	~	Titan Xp	40.1
	STDN513 [45]	DenseNet169	513 × 513	80.9	~	~	Titan Xp	28.6
	DSOD300 [46]	DS/64-192-48-1	300 × 300	77.7	14.80	59.2	Titan X	17.4
	DSOD300_lite [46]	DS/64-192-48-1	300 × 300	76.7	10.4	41.8	Titan X	25.8
	DSOD300_smallest [46]	DS/64-64-16-1	300 × 300	73.6	5.9	23.5	Titan X	~
	SqueezeNet-SSD [34]	SqueezeNet	300 × 300	64.3	5.50	~	Titan X	44.7
	MobileNet-SSD [34]	MobileNet	300 × 300	68.0	5.50	~	Titan X	59.3
	Pelee [33]	PeleeNet	304 × 304	70.9	5.43	~	TX2(FP32)	77
	Tiny DSOD [34]	G/32-48-64-80	300 × 300	72.1	0.95	~	Titan X	105
	Ours	MobileNet v2	416 × 416	81.67	10.73	47.8	RTX 2080ti	44.18

Table 6. The results of different categories on the Pascal VOC2007 test dataset.

Method	mAP	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Mbike	Person	Plant	Sheep	Sofa	Train	Tv
RON384++	77.6	86.0	82.5	76.9	69.1	59.2	86.2	85.5	87.2	59.9	81.4	73.3	85.9	86.8	82.2	79.6	52.4	78.2	76.0	86.2	78.0
SSD512	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
R-FCN	79.5	82.5	83.7	80.3	69.0	**69.2**	87.5	88.4	88.4	65.4	87.3	72.1	87.9	88.3	81.3	79.8	54.1	79.6	78.8	87.1	79.5
Faster R-CNN	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	**89.8**	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
STDN513	80.9	86.1	**89.3**	79.5	74.3	61.9	**88.5**	88.3	89.4	**67.4**	86.5	**79.5**	86.4	89.2	**88.5**	79.3	53.0	77.9	**81.4**	86.6	**85.5**
ours	**81.6**	**88.5**	87.5	**83.1**	**75.2**	67.1	85.3	**90.2**	88.9	60.9	**89.7**	78.4	**89.5**	**89.5**	84.9	**84.8**	**55.1**	**86.9**	74.3	**90.8**	82.0

Table 7. Detection results of the proposed method on the brackish dataset.

Method	mAP (%)	Big Fish (%)	Crab (%)	Jelly Fish (%)	Shrimp (%)	Small Fish (%)	Star Fish (%)	Parameters (M)	Model Size (MB)	Speed (FPS)
YOLO v4	93.56	98.57	91.39	96.86	94.77	83.96	95.82	64.04	244.0	36.91
Tiny YOLO v4	80.16	95.52	67.48	78.36	83.81	61.54	94.25	5.96	22.4	122.08
Ours	92.65	97.59	91.12	95.54	94.48	81.06	96.10	10.73	47.5	44.22

Table 8. Detection results of the proposed method on the URPC 2020 dataset.

Method	mAP (%)	Holothurian (%)	Echinus (%)	Starfish (%)	Scallop (%)
YOLO v4	81.01	71.21	89.94	85.58	77.30
Tiny YOLO v4	67.83	54.09	80.43	77.94	58.87
Ours	79.54	70.38	90.11	85.52	72.16

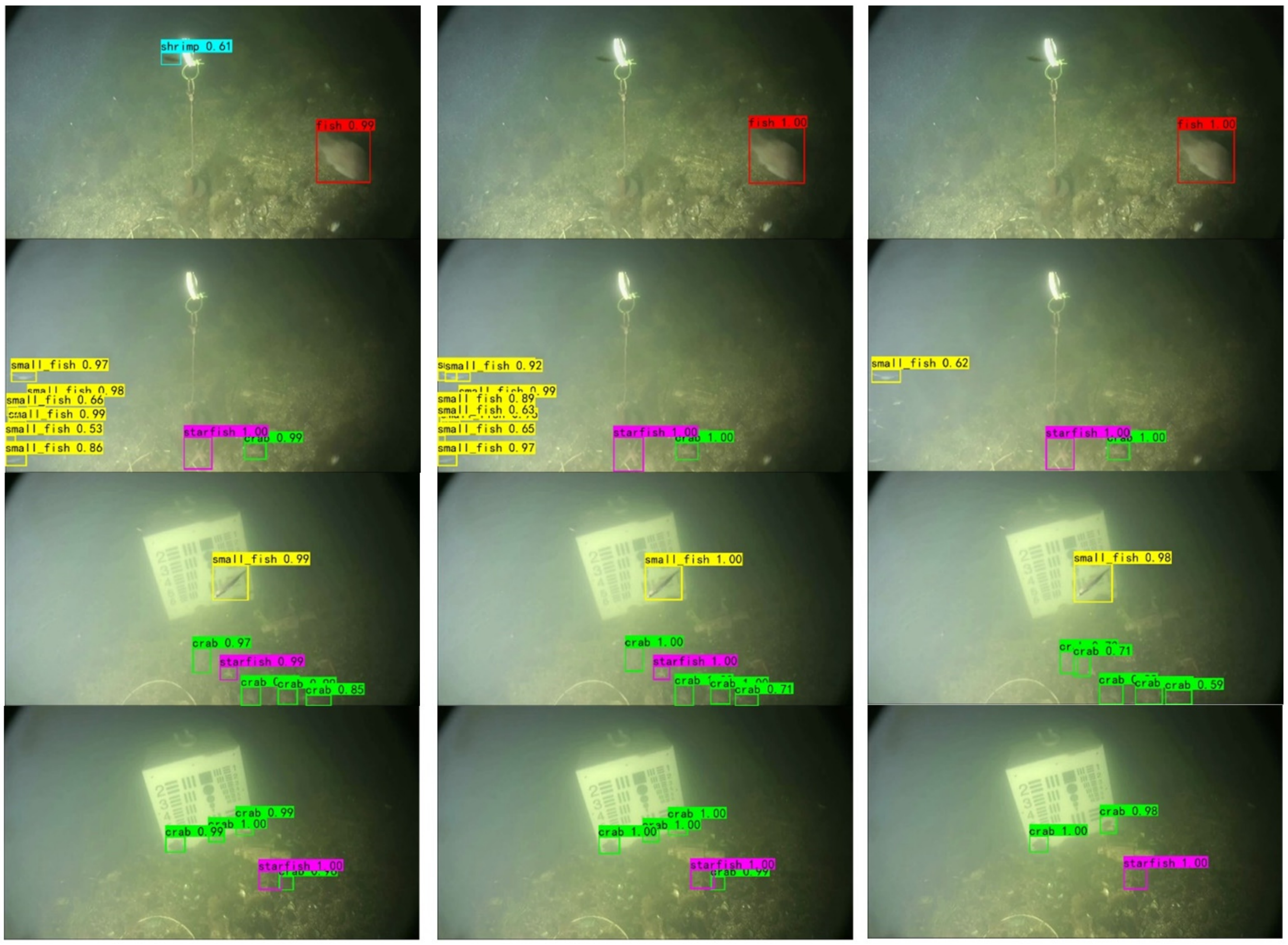

Figure 10. Visualization detection results for the brackish underwater dataset compared with YOLO v4 and Tiny YOLO v4.

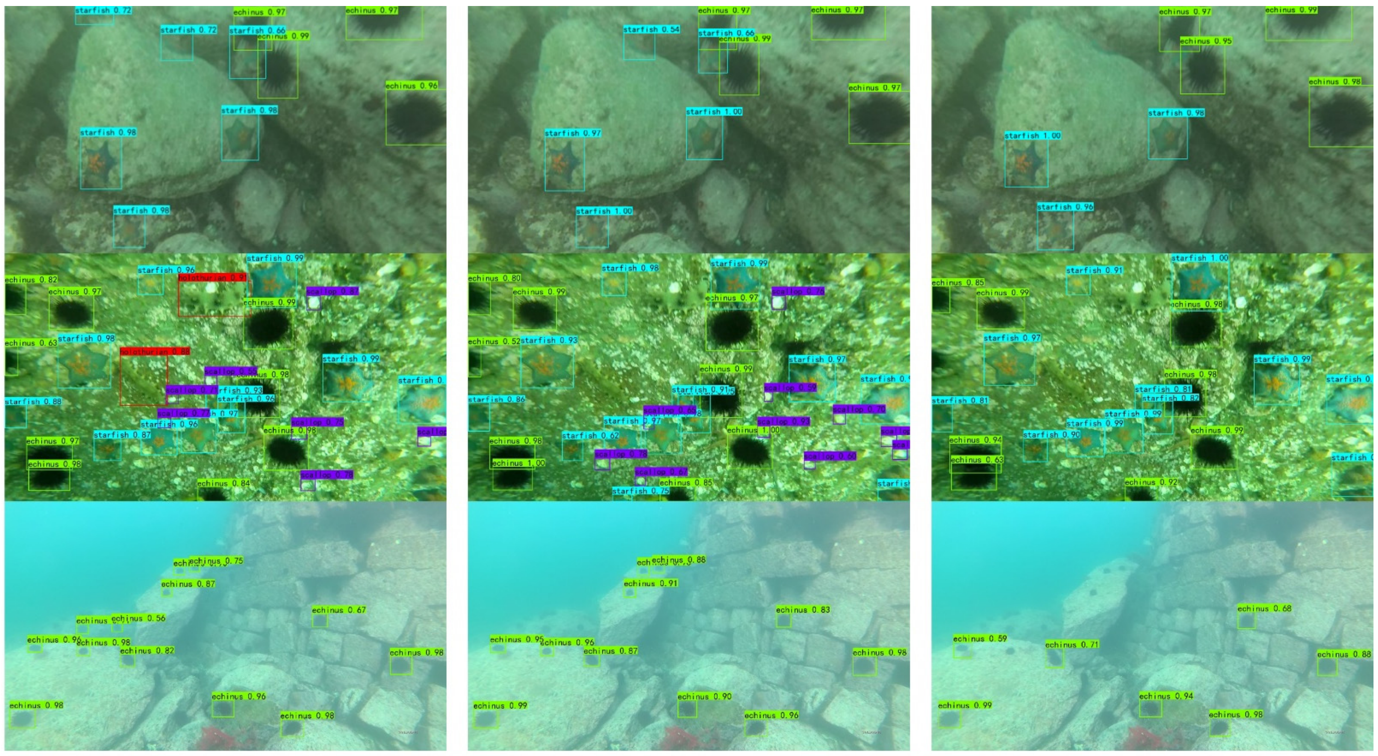

Figure 11. Visualization detection results for the URPC 2020 underwater dataset compared with YOLO v4 and Tiny YOLO v4.

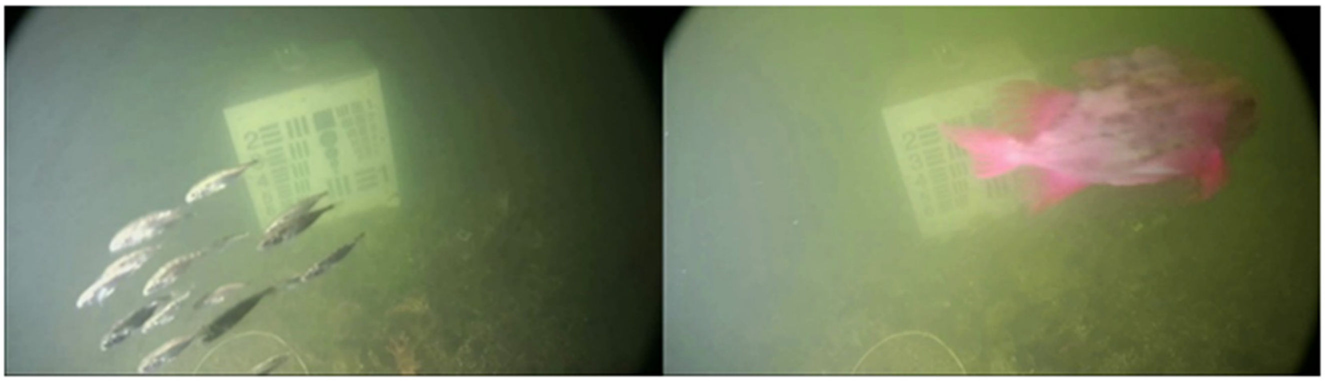

Figure 12. Original images of the brackish dataset.

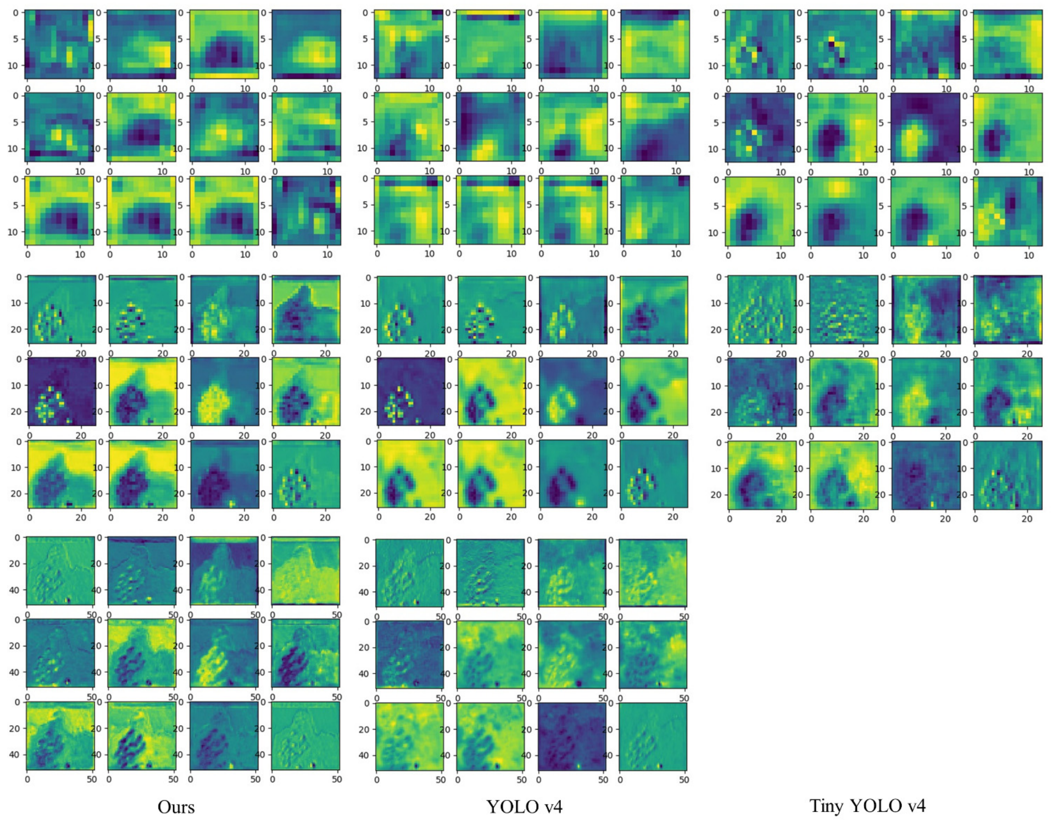

Figure 13. The feature maps of the left image in Figure 12 at different scales. Row 1: feature map of 13 × 13. Row 2: feature map of 26 × 26. Row 3: feature map of 52 × 52.

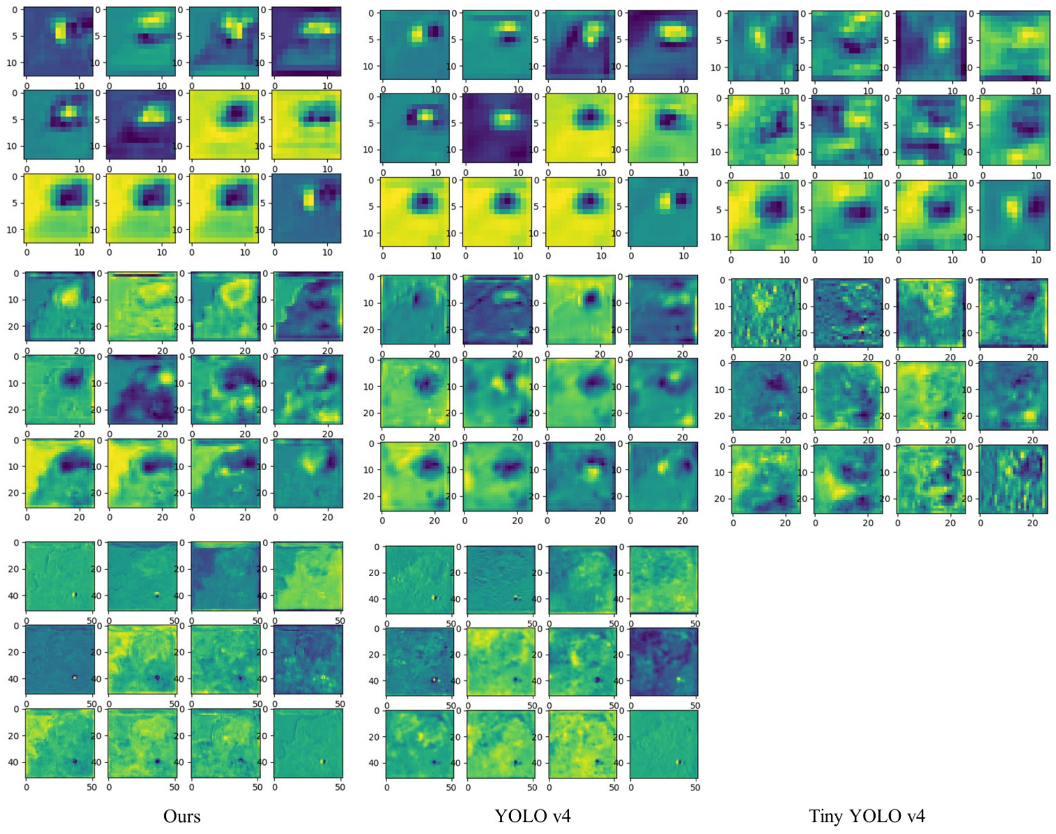

Figure 14. The feature maps of the right image in Figure 12 at different scales. Row 1: feature map of 13 × 13. Row 2: feature map of 26 × 26. Row 3: feature map of 52 × 52.

5. Discussion

5.1. Lightweight Techniques for Underwater Object Detection

Lightweight object detection is a hotspot at present, and has achieved good performance in object detection of common categories. However, there is little research on lightweight underwater object detection. Current research on lightweighting of neural network model is divided into two mainstream directions: designing lightweight networks and model compression. The study in this paper mainly focuses on building a more lightweight detection network based on YOLO v4 by combining MobileNet v2 and depth-wise separable convolution. Experiments indicate that this method can effectively reduce the number of parameters without sacrificing much accuracy. However, this kind of approach generally retains a large number of redundant parameters and channels due to the presets, even if the number of parameters is greatly reduced by changing the convolution methods. Therefore, it is necessary to prune the "least important" parameters with an appropriate criterion [47–49]. The limitation to this is that although the detector meets the requirement of real-time, the detection time (22 ms) of the proposed underwater object detector is still long, which could be further lowered. Meanwhile, the parameters can be compressed by using proper methods for applying the model into small embedded devices.

5.2. Challenges of Underwater Small Target Detection

Detection of underwater small targets is a tough problem. At present, multi-scale feature learning, data augmentation [50,51] or feature fusion are the most popular research methods for small object detection. Methods based on multi-scale feature learning and feature fusion consider both shallow texture information and deep semantic information, which is beneficial to the feature extraction of small targets. However, existing feature fusion methods such as addition and concatenation are context-independent and may lead to noise, which makes it difficult to further improve the detection performance of small targets. This paper has tried to introduce the context-dependent attention mechanisms that extract both global and local features to obtain channel attention weights, which enable the detection network to better detect underwater small targets by combining AFFM modules with FPN structures. This has confirmed that the feature fusion method based on context-dependence performs better than context-independent methods in the detection of small targets. Meanwhile, the visualization technology of CNN indicates that feature maps of different scales have different sensitivities to targets of varied scales. Unfortunately, there are still defects in our work when detecting targets under extreme scale variations. Consequently, there is still room for further research based on the feature fusion approach to better detect small targets.

5.3. Applicability in Different Underwater Marine Scenarios

In order to verify the applicability of the proposed method, we have trained and tested our method on two different underwater datasets. As it turns out, our method has achieved a similar performance compared to YOLO v4 and has reached a significantly higher accuracy than Tiny YOLO v4. However, our detector may miss detection targets when the targets are extremely small and when the features of targets are not obvious. We speculate that the extracted features are not shallow enough, despite shallow features not being conducive to classification. It is worth considering whether it is better to use shallower features for detection.

6. Conclusions

In the past, underwater object detection techniques based on optical imaging mainly focus on improving detection accuracy by using large classification networks. However, lightweight and real-time performance will perform better in practical applications. In order to solve the lightweight problem of underwater object detection, this paper proposes a lightweight underwater object detector that offers a great tradeoff between accuracy and speed. In our work, in order to build a lightweight detection network for faster and better detection of marine underwater targets, the MobileNet v2 was utilized as a backbone to extract preliminary features. Meanwhile, depth-wise separable convolution was adopted to rebuild the neck and head to reduce the number of model parameters. The improved AFFM module was designed to adapt to the FPN structure to extract features with both rich semantic information and location information, which is beneficial for underwater small targets. Experiments indicated that the parameters of the proposed network were reduced to 19.53% of YOLO v4 with a process speed of more than 44 FPS, and our detector had a similar performance on the underwater datasets compared to YOLO v4. In particular, our method achieved a 92.65% mAP and a 79.54% mAP on the brackish and URPC 2020 datasets, respectively. Furthermore, our method obtained a 81.67% mAP on the common PASCAL VOC dataset, which shows that the proposed method has certain advantages over the listed algorithms in terms of accuracy and speed.

In our future work, we will continue to further study lightweight and acceleration methods for application in marine underwater object detection. The following directions can be taken into account: (1) model pruning, (2) low-rank estimation, and (3) model distillation. In a word, the key to achieving lightweighting is to increase the compactness of the DCNN.

Author Contributions: Conceptualization, W.S.; methodology, M.Z. and S.X.; validation, S.X.; writing—original draft preparation, S.X.; writing—review and editing, M.Z., S.X., W.S., Q.H. and Q.W.; investigation, Q.W.; supervision, W.S. and Q.W.; funding acquisition, W.S. and Q.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China (61972240), and the Shanghai Science and Technology Commission part of the local university capacity building projects (20050501900).

Data Availability Statement: All data used during the study has been uploaded at https://github.com/Anleeno-Xu/Benchmark-For-Underwater-Image-Dataset (accessed on 28 April 2021).

Acknowledgments: This research was supported by the Digital Oceanography Institute of Shanghai Ocean University.

Conflicts of Interest: We declare no conflict of interest.

References

1. Deans, C.; Marmugi, L.; Renzoni, F. Active Underwater Detection with an Array of Atomic Magnetometers. *Appl. Opt.* **2018**, *57*, 2346. [CrossRef]
2. Pydyn, A.; Popek, M.; Kubacka, M.; Janowski, Ł. Exploration and Reconstruction of a Medieval Harbour Using Hydroacoustics, 3-D Shallow Seismic and Underwater Photogrammetry: A Case Study from Puck, Southern Baltic Sea. *Archaeol. Prospect.* **2021**, 1–16. [CrossRef]
3. Czub, M.; Kotwicki, L.; Lang, T.; Sanderson, H.; Klusek, Z.; Grabowski, M.; Szubska, M.; Jakacki, J.; Andrzejewski, J.; Rak, D.; et al. Deep Sea Habitats in the Chemical Warfare Dumping Areas of the Baltic Sea. *Sci. Total Environ.* **2018**, *616*, 1485–1497. [CrossRef] [PubMed]
4. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Swizerland, 2016; pp. 21–37.
5. Lin, T.-Y.; Dollar, 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 (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
6. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
7. Dai, Y.; Gieseke, F.; Oehmcke, S.; Wu, Y.; Barnard, K. Attentional Feature Fusion. In Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 5–9 January 2021; pp. 3559–3568.
8. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
9. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.

10. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
11. 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]
12. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
13. Wang, C.-Y.; Mark Liao, H.-Y.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W.; Yeh, I.-H. CSPNet: A New Backbone That Can Enhance Learning Capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
14. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
15. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
16. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
17. Pedersen, M.; HAurum, J.B.; Gade, R.; Moeslund, T.B.; Madsen, N. Detection of Marine Animals in a New Underwater Dataset with Varying Visibility. In Proceedings of the The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Long Beach, CA, USA, 15–20 June 2019.
18. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
19. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269.
20. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <0.5 MB Model Size. *arXiv* **2016**, arXiv:1602.07360.
21. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
22. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef]
23. Lin, W.-H.; Zhong, J.-X.; Liu, S.; Li, T.; Li, G. ROIMIX: Proposal-Fusion Among Multiple Images for Underwater Object Detection. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 2588–2592.
24. Uplavikar, P.; Wu, Z.; Wang, Z. All-In-One Underwater Image Enhancement Using Domain-Adversarial Learning. *arXiv* **2019**, arXiv:1905.13342.
25. Sun, X.; Shi, J.; Liu, L.; Dong, J.; Plant, C.; Wang, X.; Zhou, H. Transferring Deep Knowledge for Object Recognition in Low-Quality Underwater Videos. *Neurocomputing* **2018**, *275*, 897–908. [CrossRef]
26. Xu, F.; Wang, H.; Peng, J.; Fu, X. Scale-Aware Feature Pyramid Architecture for Marine Object Detection. *Neural Comput. Appl.* **2021**, *33*, 3637–3653. [CrossRef]
27. Pan, T.-S.; Huang, H.-C.; Lee, J.-C.; Chen, C.-H. Multi-Scale ResNet for Real-Time Underwater Object Detection. *Signal Image Video Process.* **2021**, *15*, 941–949. [CrossRef]
28. Salman, A.; Siddiqui, S.A.; Shafait, F.; Mian, A.; Shortis, M.R.; Khurshid, K.; Ulges, A.; Schwanecke, U. Automatic Fish Detection in Underwater Videos by a Deep Neural Network-Based Hybrid Motion Learning System. *ICES J. Mar. Sci.* **2019**, *77*, 1295–1307. [CrossRef]
29. Chen, L.; Liu, Z.; Tong, L.; Jiang, Z.; Wang, S.; Dong, J.; Zhou, H. Underwater Object Detection Using Invert Multi-Class Adaboost with Deep Learning. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
30. Hu, K.; Lu, F.; Lu, M.; Deng, Z.; Liu, Y. A Marine Object Detection Algorithm Based on SSD and Feature Enhancement. *Complexity* **2020**, *2020*, 5476142. [CrossRef]
31. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.
32. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
33. Wang, R.J.; Li, X.; Ling, C.X. Pelee: A Real-Time Object Detection System on Mobile Devices. *arXiv* **2019**, arXiv:1804.06882.
34. Li, Y.; Li, J.; Lin, W.; Li, J. Tiny-DSOD: Lightweight Object Detection for Resource-Restricted Usages. *arXiv* **2018**, arXiv:1807.11013.
35. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556.
36. Misra, D. Mish: A Self Regularized Non-Monotonic Activation Function. *arXiv* **2020**, arXiv:1908.08681.
37. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
38. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
39. Loshchilov, I.; Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv* **2017**, arXiv:1608.03983.

40. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switerland, 2014; Volume 8693, pp. 740–755. ISBN 978-3-319-10601-4.
41. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y.M. Scaled-YOLOv4: Scaling Cross Stage Partial Network. *arXiv* **2021**, arXiv:2011.08036.
42. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object Detection via Region-Based Fully Convolutional Networks. *arXiv* **2016**, arXiv:1605.06409.
43. Singh, B.; Li, H.; Sharma, A.; Davis, L.S. R-FCN-3000 at 30fps: Decoupling Detection and Classification. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1081–1090.
44. Kong, T.; Sun, F.; Yao, A.; Liu, H.; Lu, M.; Chen, Y. RON: Reverse Connection with Objectness Prior Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5244–5252.
45. Zhou, P.; Ni, B.; Geng, C.; Hu, J.; Xu, Y. Scale-Transferrable Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 528–537.
46. Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.-G.; Chen, Y.; Xue, X. DSOD: Learning Deeply Supervised Object Detectors from Scratch. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Honolulu, HI, USA, 21–26 July 2017; pp. 1937–1945.
47. Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv* **2016**, arXiv:1510.00149.
48. Molchanov, P.; Tyree, S.; Karras, T.; Aila, T.; Kautz, J. Pruning Convolutional Neural Networks for Resource Efficient Inference. *arXiv* **2017**, arXiv:1611.06440.
49. He, Y.; Zhang, X.; Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1398–1406.
50. Kisantal, M.; Wojna, Z.; Murawski, J.; Naruniec, J.; Cho, K. Augmentation for Small Object Detection. In Proceedings of the 9th International Conference on Advances in Computing and Information Technology (ACITY 2019), Sydney, Australia, 21–22 December 2019; Aircc Publishing Corporation: Chennai, India, 2019; pp. 119–133.
51. Li, J.; Liang, X.; Wei, Y.; Xu, T.; Feng, J.; Yan, S. Perceptual Generative Adversarial Networks for Small Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1951–1959.