Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 15, No. 2, 2024

Improved YOLOV7-TINY Network for Sea Bream Detection

Linhua Jiang[1, 2], Yuanyuan Yang[1], Entuo Liu[2], Lingxi Hu[1], Jiahao Xu[1], Lei Chen[1], Jintao Zhang[1], Peng Liu[1], Wei Long[1, *]

[1] School of Information Engineering, Huzhou University, Huzhou 313000, China
[2] The Institute of Advanced Vision Research, Hangzhou Research Institute of Xidian University, Hangzhou 311231, China

[*] **Corresponding author**: Wei Long (Email: 1w@zjhu.edu.cn)

Abstract: Accurate identification of underwater fish species is of great scientific and economic significance in aquaculture, as it can provide scientific basis for aquaculture production and promote related research. However, the complexity of the underwater environment is affected by various factors such as light, water quality, and mutual occlusion of fish species. Therefore, underwater fish images are often not clear enough, which limits the accurate identification of underwater targets. In this paper, an improved YOLOV7-TINY model for sea bream detection is proposed. We employ FasterNet to replace the backbone network of the YOLOV7-TINY model, further reducing model parameters and computational complexity without compromising accuracy. By leveraging cascaded feature fusion in the backbone network, we effectively address the challenges posed by multi-scale datasets and insufficient information extraction. Additionally, the RESNETCBAM attention mechanism is incorporated into the feature maps at three different scales, allowing the network to better capture relevant information from complex underwater environments while minimizing unnecessary interference. Finally, the ECIOU loss function is adopted to optimize frame adjustments and reduce the training time of the model.

Keywords: Cascaded feature fusion; Attention mechanism; Improved YOLOV7-TINY network; ECIOU.

1. Introduction

In recent years, with the rapid development of deep learning technology and its successful applications in various fields such as facial recognition, autonomous driving, speech recognition, and text processing , the structural system of deep learning technology has become more mature and perfected. Leveraging its powerful data processing capabilities, an increasing number of researchers have begun to apply it to the field of aquaculture. This trend not only drives further development of deep learning technology but also brings tremendous convenience to aquaculture work, further promoting the development of the aquaculture industry. For aquaculture enterprises, applying deep learning technology to aquaculture work is of great significance. Through deep learning technology, individual recognition of cultured fish in fixed waters can better grasp the growth status of cultured fish. Moreover, it enables the formulation of more reasonable feeding plans based on the overall number of cultured fish, facilitating precise feeding and reducing economic losses caused by improper feeding to some extent. Additionally, individual recognition of fish can enable aquaculture workers to promptly understand the health status of cultured fish, allowing timely treatment when they are sick, thus ensuring better growth of cultured fish and reducing unnecessary economic losses for aquaculture enterprises.

Because of the underwater environment and numerous other uncertain factors, individual fish recognition still faces many challenges. In the development process of fish individual recognition, it is mainly divided into traditional fish individual recognition and deep learning-based individual recognition methods. Traditional fish individual recognition requires manual feature extraction, which is time-consuming and inefficient, resulting in lower recognition accuracy for different fish species. Strachan et al. established and classified databases of different types of fish photos using computers. Nathalie Castignolles et al. observed and counted the number of fish species passing through rivers through window glass. Lee et al. used holistic shape matching for fish recognition, which improved the accuracy of fish individual recognition, but the high computational cost led to limited practicality. In traditional machine learning, fish recognition work is completed by computers instead of humans through analyzing the acquired data. Ding Shunrong et al. used a fish classification method based on particle swarm optimization SVM, which can classify fish more accurately compared to traditional support vector machines, but the recognition accuracy decreases when dealing with similar morphologies and textures of the same species. Yao Runlu et al. used digital image processing and BP neural networks for freshwater fish recognition, but their innovation in methodology is still limited. Petr Cisar et al. introduced a method for the automatic individual recognition of Atlantic salmon based on dot patterns on the skin. Renato et al. utilized photo recognition technology for individual identification of cichlid fish.

Traditional object detection algorithms have been constrained by many limitations, including poor robustness and difficulty in feature extraction. In contrast, deep learning neural networks can map low-dimensional image data to high-dimensional space, automatically learn feature mappings through operations such as convolution and pooling during forward propagation. Following AlexNet and VGG , many excellent deep learning frameworks have been proposed, such as Recurrent Neural Networks (RNN) , Long Short-Term Memory (LSTM) , Generative Adversarial Network (GAN) , GoogLeNet , ResNet , MobileNet series , , , and DenseNet , etc. Models based on convolutional neural networks utilize parameter sharing, speeding up the computation of algorithms. Based on this, many researchers

have adopted this idea to construct object detection algorithms. Currently, popular object detection algorithms can be categorized into two types: two-stage and one-stage. The former is based on the principle of coarse positioning and fine classification, first identifying candidate regions containing objects and then performing classification, such as R-CNN , Fast-RCNN , Faster-RCNN , Mask-RCNN , etc., which are relatively slower in detection speed compared to the latter. One-stage object detection algorithms directly predict object classification and localization through convolutional neural networks, achieving a better balance between accuracy and speed. Representative algorithms in this category include SSD , the YOLO series (YOLOV3 , YOLOV4 , YOLOV5 , YOLOV6 , YOLOV7). Zhao et al. proposed a YOLO-UOD underwater detection algorithm based on Yolov4-tiny. Li et al. introduced a triplet attention mechanism in YOLOV5 to improve underwater biological feature extraction capabilities. Zhai et al. added CBAM to Yolov5s to improve recognition accuracy and efficiency and introduced a multi-scale algorithm to enhance image contrast.

Due to the complex underwater environment, simply applying the above models to sea bream recognition still poses some problems:

(1) Underwater environments are affected by factors such as lighting and water quality, and the background of the captured images also poses difficulties for detection, leading to inaccurate detection results.

(2) During the process of feature extraction and fusion, the models may not fully extract multi-scale information from underwater fish schools.

(3) The original model has long training times and is large in size, making it inconvenient for deployment on mobile devices.

Our work aims to address the aforementioned issues.

2. Methods

2.1. FasterNet

With the presence of significant redundant computations in the backbone network of YOLOV7-TINY, FasterNet is based on reducing redundant computations and memory access, which can further improve accuracy without compromising the original Baseline.

For lightweight networks such as MobileNet, ShuffleNet, and GhostNet, they utilize depthwise convolution (DWConv) or group convolution (GConv) to extract spatial features. However, in the process of reducing floating-point operations (FLOPs), as indicated by (1), detection latency may also be influenced by the number of floating-point operations per second (FLOPS), and operators are often affected by the side effects of increased memory access.

$$Latency = \frac{FLOPs}{FLOPS} \quad (1)$$

To achieve a high number of floating-point operations per second (FLOPS) while reducing FLOPs, partial convolution (PConv) is used to decrease memory access and computational redundancy. Essentially, the FLOPs of PConv are lower than regular Convolution, but FLOPS are higher than DWConv/GConv. In other words, PConv better utilizes the computational power of the device, while also being effective in extracting spatial features.

In Figure 1, PConv applies regular Convolution to extract spatial features from some input channels while keeping the rest unchanged. For contiguous or regular memory access, the first or last contiguous c_p channels are considered as representatives of the entire feature map for computation. Without loss of generality, it is assumed that the input and output feature maps have the same number of channels.

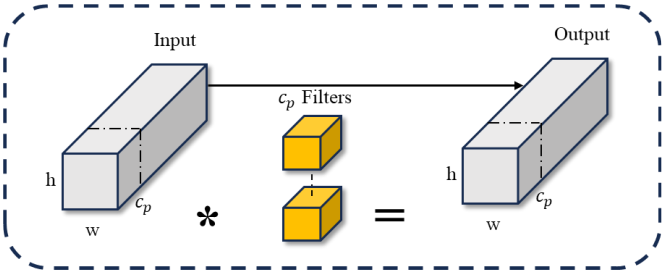

Figure 1. PConv structure

The FLOPs (Floating Point Operations) of PConv can be expressed as $h \times w \times k^2 \times c_p^2$,where h and w represent the width and height of the feature map, k denotes the size of the convolution kernel, and c_p signifies the number of channels in the conventional convolution. In PConv, c_{in} in conventional convolution is replaced by c_{P}. In practical applications, $r = \frac{c_p}{c} = \frac{1}{4}$ is typically assumed. Consequently, the FLOPs of PConv are only 1/16 of those of conventional convolution.

The memory access pattern of PConv can be represented as $h \times w \times 2c_p + k^2 \times c_p^2$, which is approximately $h \times w \times 2c_p$, where h and w are the width and height of the feature map, k is the size of the convolution kernel, and c_{P} is the number of channels in the conventional convolution. The memory access of PConv is only a quarter of that of conventional convolution, hence no additional memory access is required.

In order to fully and effectively utilize information from all channels, a pointwise convolution (PWConv) is added after the PConv. The FLOPs of PWConv and PConv decoupling can be calculated as $h \times w \times (k^2 \times c_p^2 + c \times c_p)$.Compared to regular convolutions, this reduces the computational workload. Each FasterNet module consists of one PConv layer, two PWConv layers, batch normalization, and a ReLU activation function, as shown in Figure 2.

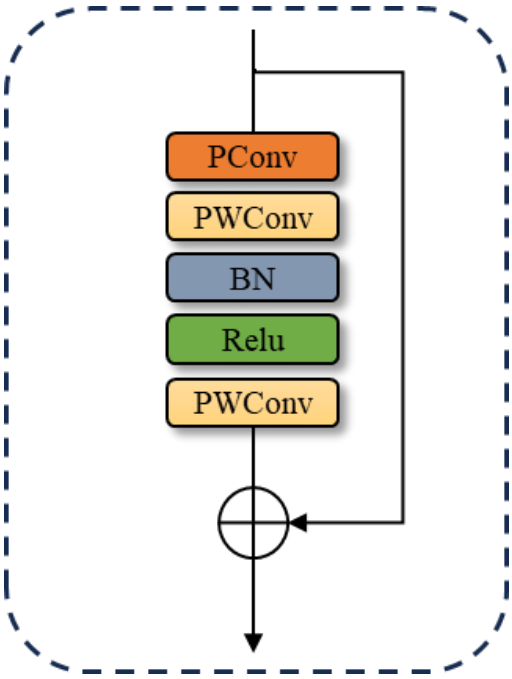

Figure 2. FasterNet Block

2.2. Fussion Block

The Fusion Block achieves more accurate object detection and localization by integrating feature maps from different levels. Specifically, it consists of two essential components: the feature fusion module and the multi-scale feature fusion module. The feature fusion module merges feature maps from different levels through a feature pyramid network, thereby enhancing the model's capability for object detection. On the other hand, the multi-scale feature fusion module combines feature maps from different scales, enabling the model to better adapt to objects of varying sizes and proportions. The combination of these two modules leads to improved handling of complex environmental conditions and other challenges.

Specifically, this involves reducing the dimensionality of feature maps at the same scale using 1x1 convolutions. For larger-scale feature maps, dimensionality reduction is performed initially using 1x1 convolutions, followed by downsampling using a 3x3 convolution with a stride of 2. For smaller-scale feature maps, upsampling is conducted using 2x2 transpose convolutions. Finally, the resulting feature maps from these three parts are concatenated together, followed by dimensionality reduction using 1x1 convolutions again, as shown in Figure 3.

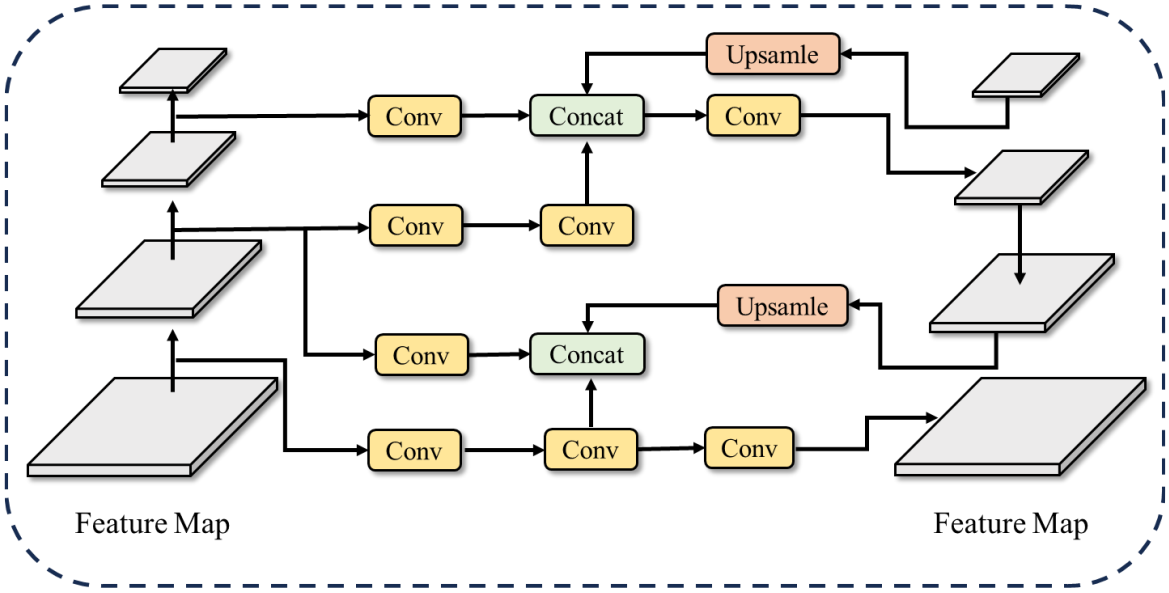

Figure 3. Fussion Block

2.3. ResNetCBAM

The attention mechanism in deep learning is a mechanism that promotes autonomous learning in network models and captures important features. The Convolutional Block Attention Module (CBAM) is a simple and effective attention module. It concatenates channel attention and spatial attention modules between the input and output of the CBAM structure. Both of these modules use global max-pooling and global average-pooling to extract richer global and local semantic information. CBAM first calculates its channel attention features from the channel dimension and then multiplies these channel attention features with the original feature maps, recalculating the channel weights of the original feature maps. Subsequently, the feature maps with channel attention are input into the spatial attention module, allowing the network to adaptively learn the importance of different pixel positions within the same channel, ultimately obtaining filtered important features.

For spatial attention, it is based on global average pooling and global max pooling along the channel dimension to obtain two feature maps with dimensions HxWx1. These two feature maps are concatenated along the channel dimension to form a feature map with dimensions HxWx2. Subsequently, the concatenated feature map undergoes a 7x7 convolution to reduce the channel dimension to 1. Finally, the output is obtained through the sigmoid activation function. The spatial attention mechanism is illustrated in Figure 4.

For channel attention, global max pooling and global average pooling are utilized to integrate spatial information. Max pooling retains the most salient features within the pooling region, aiming to extract prominent characteristics of the target while preserving texture, contours, and other details in the image to the fullest extent. On the other hand, average pooling computes the average value of all elements within the region, retaining more background information of the image. The specific steps for obtaining channel attention are as follows: perform max pooling and average pooling along the spatial dimension for an input with dimensions HxWxC, resulting in two features with dimensions 1x1xC each. These features are then fed into a shared network consisting of multi-layer perceptrons (MLP) with hidden layers. After computation, the output feature maps are added element-wise, and the final channel feature attention is obtained using the sigmoid activation function. The channel attention mechanism is illustrated in Figure 5.

ResNetCBAM is an extension of the ResNet architecture, where the attention module CBAM is concatenated after the convolution within the residual block and before the shortcut connection. This addition aims to filter out effective features from the feature maps. The CBAM module operates by redistributing the weights of the extracted feature maps without altering the original ResNet network structure. By extracting both channel and spatial attention, the residual block not only pays attention to whether there are target tasks but also where the targets are located. This effectively enhances the feature extraction capability of the network model. Figure 6 illustrates the schematic diagram of the residual structure after adding the CBAM module.

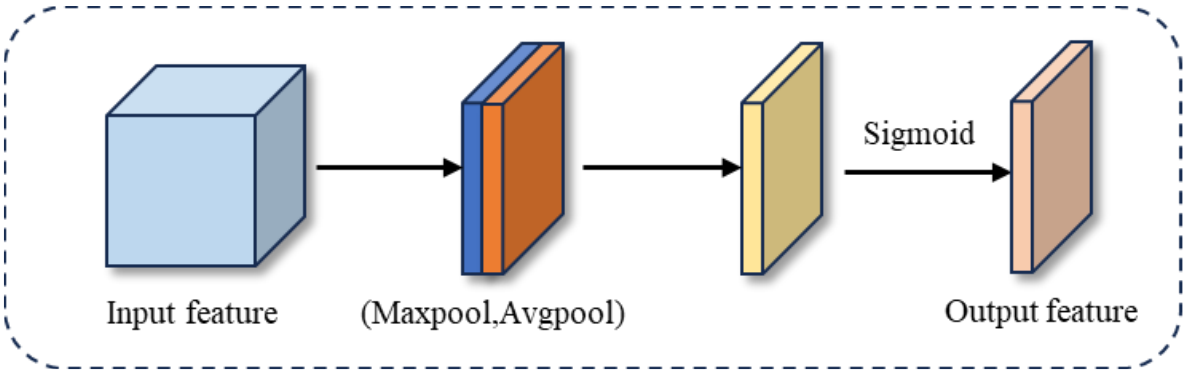

Figure 4. Spatial Attention Module

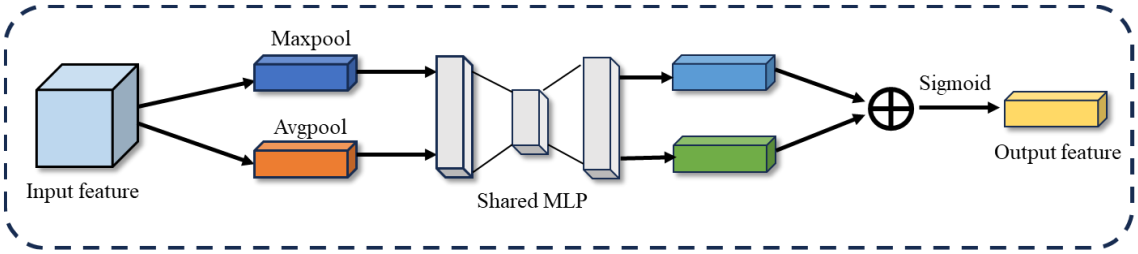

Figure 5. Channel Attention Module

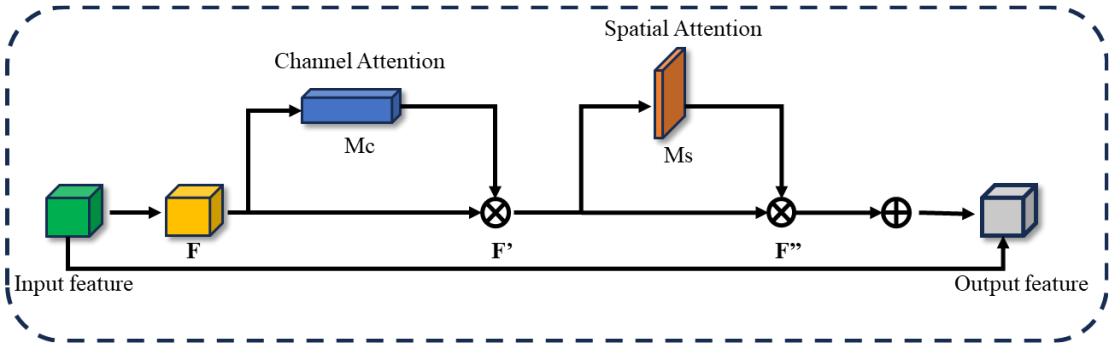

Figure 6. ResNetCBAM

2.4. Loss Function

Object detection can typically be divided into two stages: localization and detection. The accuracy of the localization stage is primarily influenced by regression loss functions, leading to the emergence of various new regression loss functions.

To measure the similarity between predicted boxes and ground truth boxes and to select appropriate positive and negative samples, Intersection over Union (IoU) has become the most popular metric in bounding box regression. To further optimize the IoU metric, the IoU loss function has been proposed.

However, the IoU loss function fails when there is no overlap between the predicted box and the IoU loss function. To address these issues, many IoU-based evaluation systems have been derived, which improve the shortcomings of the original IoU loss function from different perspectives, significantly enhancing its robustness.

Among them, the Generalized Intersection over Union (GIoU) , Distance Intersection over Union (DIoU) , and Complete Intersection over Union (CIoU) loss functions are the most representative methods. They have made significant progress in the field of object detection, but there is still considerable room for optimization.

Among these methods, CIoU is considered the most optimal boundary regression loss function because it considers three key geometric factors: overlap area, center point distance, and aspect ratio. CIoU utilizes IoU, Euclidean distance, corresponding aspect ratio, and angle to measure the overlap area between the target and the ground truth box.

In the regression stage, it is not suitable for both width and height of the predicted box to increase or decrease simultaneously because αv cannot accurately represent the confidence in the true width and height differences. Therefore, when the model converges to the linear ratio between the width and height of the predicted frame and the true frame, it may sometimes hinder the effective optimization of similarity. The EIOU_Loss function decomposes the aspect ratio factor αv in CIOU_Loss to calculate the width and height of the predicted frame and the true frame, thus addressing the problem of CIOU_Loss.

When dealing with distant edges, only the calculation of EIOU_Loss can become slow and may not converge prematurely. To address this issue, a new enhanced loss function called ECIOU has been proposed, which can facilitate adjustments to the predicted frame and improve frame regression rates.

The foundation of ECIOU is the combination of the CIOU and EIOU loss functions. Initially, the aspect ratio of the predicted frame is adjusted by CIOU until it converges to an appropriate range. Then, each side is finely tuned by EIOU until it converges to the correct value.ECIOU_Loss is

computed using Equation (5).

$$\mathcal{R}_{CIoU} = \frac{\rho^2(\boldsymbol{b}, \boldsymbol{b}^{gt})}{a^2} + \alpha v \tag{2}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \tag{3}$$

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2(\boldsymbol{b}, \boldsymbol{b}^{gt})}{a^2} + \alpha v \tag{4}$$

$$ECIOU_{Loss} = 1 - IOU + \alpha v + \frac{\rho^2(b^{gt}, b)}{c^2} + \frac{\rho^2(h^{gt}, h)}{c_h^2} + \frac{\rho^2(w^{gt}, w)}{c_w^2} \tag{5}$$

2.5. The improved YOLOv7-TINY network architecture

The YOLO (You Only Look Once) network is a popular real-time object detection algorithm. Its main idea is to treat the object detection problem as a single regression problem, by directly predicting the positions and classes of bounding boxes in the neural network. Due to the simple and efficient design of the YOLO series, it has become one of the preferred algorithms for real-time object detection tasks.

By utilized YOLOv7-TINY as the BASELINE and replaced the backbone network with the FasterNet network, we aim to reduce redundant computations without sacrificing accuracy. Subsequently, we employed the multi-scale BIFUSSION to better integrate contextual information. Before the prediction head, we separately added theRESNETCBAM attention mechanism after the ELAN module, allowing the network to better capture and represent important features, thus enhancing the model's performance and generalization ability. The YOLOv7-TINY (BASELINE) and the improved ResNetCBAM-FUSSION-YOLO network architecture are illustrated in Figures 7 and 8, respectively.

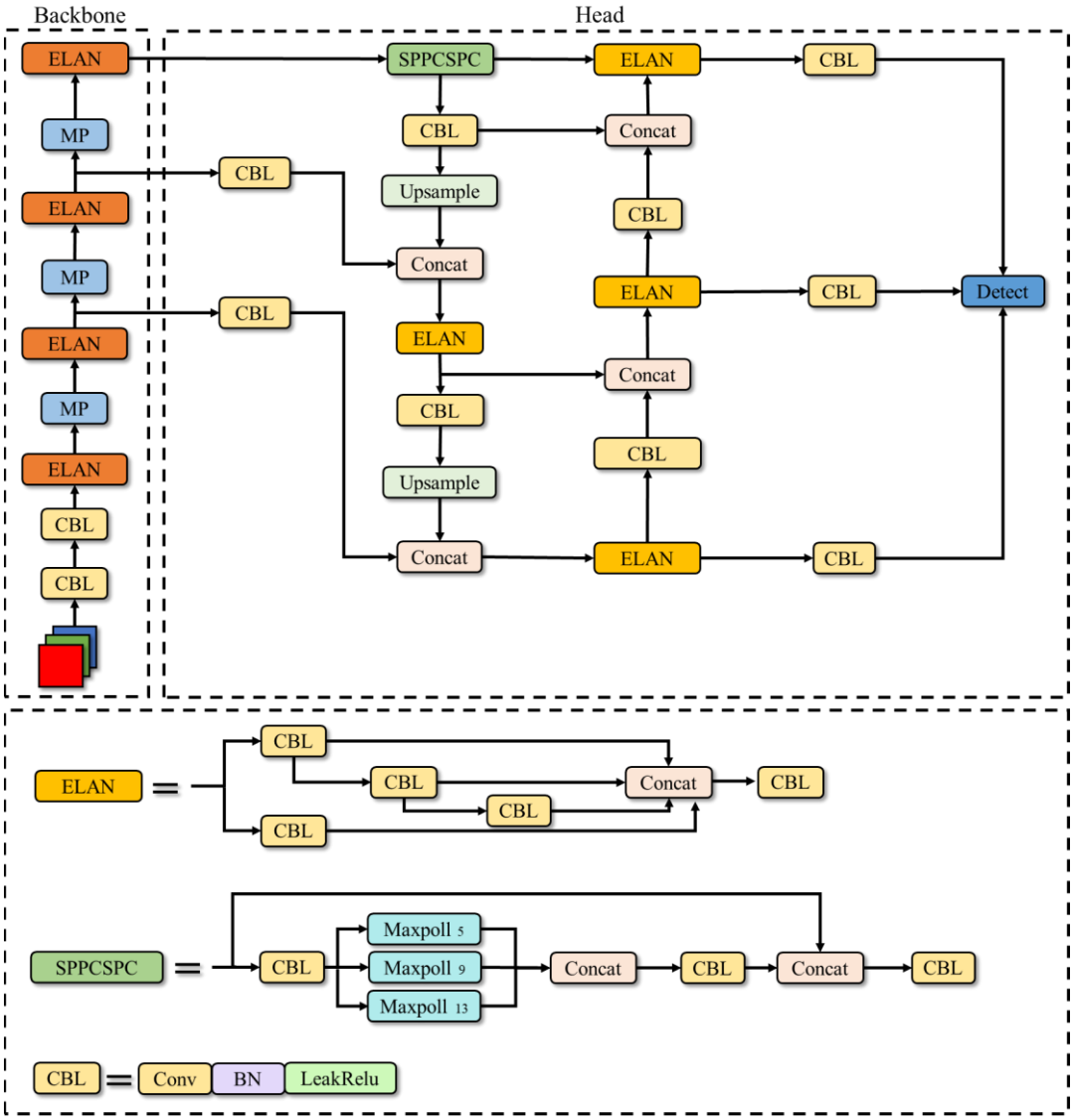

Figure 7. YOLOV7-TINY

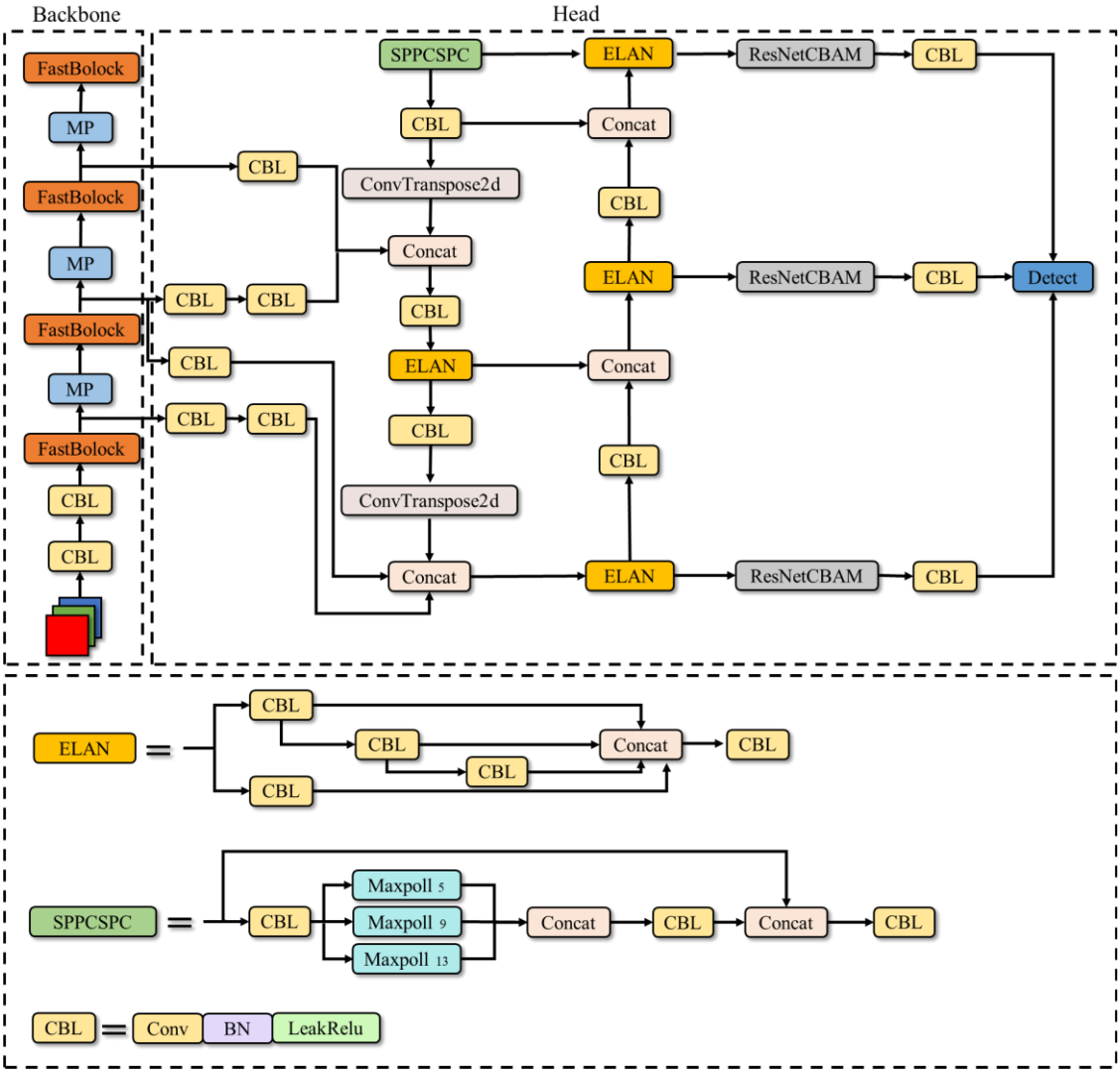

Figure 8. ResNetCBAM-FUSSION-YOLO

3. Results and Analysis

3.1. Environment configuration

When training the model, the LINUX operating system was utilized alongside the PyTorch 1.12.0 framework. The server configuration included an Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz processor, 188GB of memory, and an NVIDIA A100 PCIe GPU with 80GB of memory, running CUDA 12.2 driver version. The input image size was set to 640x640, with a batch size of 16 and 200 training steps. The learning rate was set to 0.01, momentum to 0.937, and the weight decay for stochastic gradient descent (SGD) was 0.0005. For training and testing purposes, the PyCharm software was installed on a local WINDOWS system, communicating with the server via remote connection. The original images were divided into training, validation, and test sets in an 8:1:1 ratio. Additionally, data augmentation was applied to the training set to enhance the robustness of the experiment.

3.2. Model evaluation criteria

In this experiment, a comprehensive set of metrics was adopted to evaluate the performance of fish object detection.

Precision(P) represents the proportion of positive samples in the samples with positive prediction results. The definition is shown in Equation (6):

$$P = \frac{TP}{TP + FP} \tag{6}$$

Recall(R) represents the prediction result as the proportion of the actual positive samples in the positive samples to the positive samples in the whole sample. The calculation formula is as follows:

$$R = \frac{TP}{TP+FN} \tag{7}$$

AP (average precision) refers to the average accuracy in object detection. It combines the model's performance under different precision and recall conditions and reflects the balance between accuracy and recall. AP is calculated as the area under the precision–recall curve (PR curve), as shown in Equation (8).

$$AP = \int_0^1 P(R)dR \tag{8}$$

The mAP metric evaluates the overall performance of the model across all categories. It is calculated by taking the average of the AP (Average Precision) values for different categories, as shown in Equation (9):

$$mAP = \frac{\sum_{i=0}^{n} AP(i)}{n} \tag{9}$$

3.3. Experiment

3.3.1. Data Preparation

The data images were collected from Időkép and

manually annotated using LabelImg software for 1430 images with a resolution of 1920x1080. To ensure the diversity and completeness of the data, we selected images from multiple different time periods for capture. Through manual selection, we ensured that the images have complete boundaries and necessary clarity to guarantee the quality of the data. Data augmentation was applied to the training set to enhance the robustness of the experiment.

3.3.2. Analysis of the results of the ResNetCBAM-FUSSION-YOLO experiment

In this experiment, a batch size of 16 was used, meaning that 16 images were processed in each training iteration. The YOLOV7 network provides good visualization effects. After the model training is completed, test_batch_label.jpg is generated to observe the real bounding boxes on the validation set for that batch, and test_batch_pred.jpg is generated to observe the predicted bounding boxes on the validation set for that batch. The predicted results on the validation set during model training are shown in Figure 9.

Figure 9. Validation set results

3.3.3. Comparison of loss function results of different IOU

According to Table 1, we employed four loss functions, namely SIOU, EIOU, CIOU, and ECIOU, on the same model, and compared their training times. As indicated by the graph, the training time for ECIOU is approximately 1.1 hours, which is notably shorter compared to the other three methods for completing 200 epochs. This would significantly enhance practical training efficiency. Additionally, as shown in Table 2, in the ResNetCBAM-FUSSION-YOLO model using the ECIOU loss function, the mAP@0.5 is higher by 1.2%, 2.6%, and 1.1% compared to using the SIOU, CIOU, and EIOU loss functions, respectively. Moreover, the mAP0.5:0.95 is higher by 3.3%, 3%, and 1.7% compared to using the SIOU, CIOU, and EIOU loss functions, respectively. Both precision and recall also perform the best with the ECIOU loss function among these four. Consequently, we selected ECIOU as the loss function for the ResNetCBAM-FUSSION-YOLO network model.

In summary, the improved YOLOv7-TINY model has better detection accuracy compared to the original model, with the MAP@0.5 increasing from 90.6% to 92.6%, indicating that the proposed network model offers higher detection precision. The computational complexity is reduced by 16.7%, the number of parameters decreases by 11.5%, and the model training time is shortened by 49.2%. This reduction in hardware resource requirements facilitates model deployment in mobile devices and provides a reference for intelligent aquaculture applications.

Table 1. Loss function and training time

IOU	Training Time/h
SIOU	1.896
EIOU	1.61
CIOU	1.893
ECIOU	1.113

Table 2. Test results for different loss functions

MODEL	MAP0.5/%	MAP@.5:0.9/5%	P/%	R/%
SIOU	91.4	49.5	91.7	85.9
CIOU	90	49.8	92.1	85.5
EIOU	91.5	51.1	92.1	85.7
ECIOU	92.6	52.8	94.1	86.2

3.3.4. Comparative Experiments of Various Mainstream Models

To demonstrate the superior performance of the proposed model on lightweight devices in the detection of fish schools, a comparison of the improved network proposed in this paper with mainstream target recognition networks during detection is shown in Table 3. In Table 3, our model P and R are both superior to the other algorithms listed in the table, indicating good model performance. In terms of mAP@0.5, our method shows improvements of 3.5% and 9.3% over SSD and Faster-RCNN, respectively. Furthermore, our model's performance surpasses YOLOV3, YOLOV4, and YOLOV5s by 1.8%, 3.4%, and 2.3%, respectively. In terms of mAP@0.5:0.95, it surpasses SSD and Faster-RCNN by 2.3% and 6.2%,

respectively, and surpasses YOLOV4 and YOLOV5s by 1% and 0.8%, respectively. Our model's computational and parameter quantities are the smallest in the table, only 11 and 5.32M, respectively, meaning that it consumes minimal hardware resources in device deployment while maintaining high detection accuracy. Additionally, in terms of detection speed, our model achieves 85.9FPS, surpassing the inference speeds of other mainstream networks. Figure 10 compares mAP@0.5 and FPS under different models.

Table 3. Comparative Experiment

MODEL	MAP@0.5/%	MAP@.5:0.95/%	FPS	P/%	R/%	Parameters/M	GFLOPs/G
YOLOV5s	90.3	52	58.62	93.2	85.5	7.01M	15.8
YOLOV4	89.2	51.8	40	92.6	84.4	9.11M	20.6
YOLOV3	90.8	53.8	49.4	93.3	85.3	6.15M	154.5
Faster-RCNN	83.3	46.6	38.7	67	85.3	28.47M	470.1
SSD	89.1	50.5	47.1	90	83.8	26.15M	31.4
Our model	92.6	52.8	85.9	94.1	86.2	5.32M	11

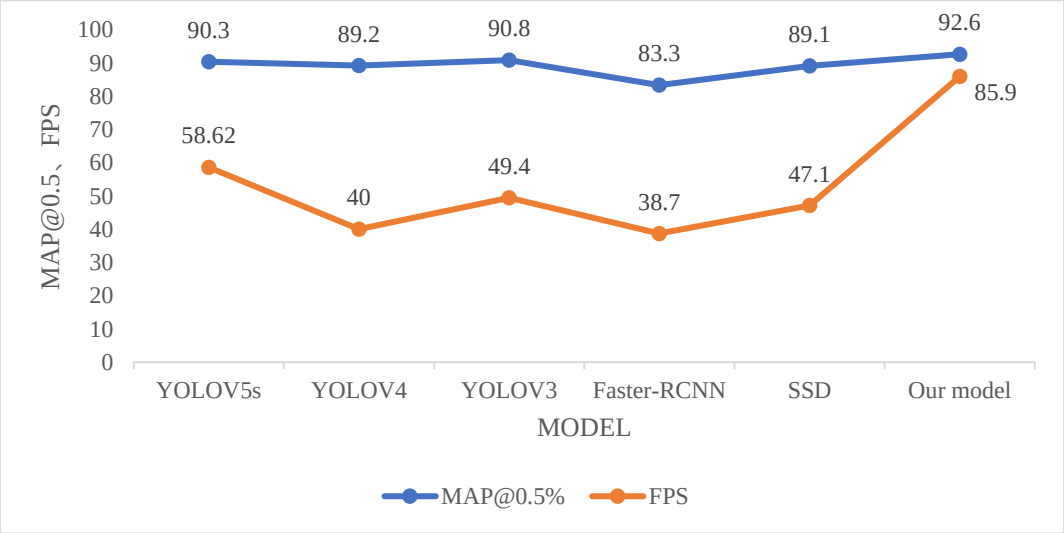

Figure 10. Comparison of map@0.5 and FPS under different models

3.3.5. Ablation experiment

To validate the effectiveness of each module in the improved network, we conducted ablation experiments. We compared our proposed ResNetCBAM-FUSSION-YOLO model with several counterparts, including the original YOLOV7, YOLOV7-TINY, YOLOV7-TINY with FasterNet backbone (YOLOV7-TINY-FasterNet), YOLOV7-TINY with FasterNet backbone and cascade context fusion module (YOLOV7-TINY-FasterNet-Fussion), and ResNetCBAM-FUSSION-YOLO with RESNETCBAM attention mechanism, in terms of GPU consumption, mAP values, parameter count, accuracy, recall, and computational complexity.According to the analysis in Table 4, it can be observed that our proposed ResNetCBAM-FUSSION-YOLO model occupies 14.58% of the parameters of the original YOLOV7 model, reducing parameter count by 0.69 million compared to the YOLOV7-TINY model. With the same batch size, ResNetCBAM-FUSSION-YOLO consumes only 3.8% of GPU, which is only 0.9% higher than the baseline, while the YOLOV7 model consumes 51.3%. In terms of MAP@0.5 evaluation, ResNetCBAM-FUSSION-YOLO outperforms YOLOV7-TINY by 2 percentage points, surpassing YOLOV7-TINY-FasterNet-Fussion and YOLOV7-TINY-FasterNet by 0.8 and 1.4 percentage points respectively. In terms of computational complexity, it reduces by 2.2 compared to YOLOV-TINY.It is worth noting that the ResNetCBAM-FUSSION-YOLO model achieves higher accuracy and requires fewer hardware resources compared to YOLOV7-TINY. Our model also shows an improvement in FPS compared to the baseline. Furthermore, although YOLOV7 exhibits higher accuracy, it has slower detection speed and higher hardware resource requirements compared to other models, making it challenging to deploy in practical scenarios.

Table 4. Ablation results

MODEL	GPU/%	MAP@0.5/%	MAP@.5:0.95/%	PARAMETERS /M	P/%	R/%	GFLOPs/G	FPS
YOLOV7	51.3	93.8	56.5	36.5	93.6	88.6	103.8	52.4
YOLOV7-TINY	2.9	90.6	51.6	6.01	91	87.3	13.2	79.9
YOLOV7-TINY-FasterNet	3.1	91.2	50.1	4.28	90.9	86.6	9	95.6
YOLOV7-TINY-FasterNet-Fussion	3.2	91.8	51.5	4.29	91.8	86	9.1	90.7
Our model	3.8	92.6	52.8	5.32	94.1	86.2	11	85.9

3.3.6. ResNetCBAM-FUSSION-YOLO test results

After multiple iterations of deep learning network training and fine-tuning on the validation set, we selected the best models and loaded them into the computer to obtain inference

results on the test set. Among them, we selected three models, namely ResNetCBAM-FUSSION-YOLO, YOLOV7, and YOLOV7-Tiny, to detect fish schools, as shown in Figures 11, 12, and 13. It can be observed that compared to YOLOV7 and YOLOV7-Tiny, ResNetCBAM-FUSSION-YOLO can detect all fish schools in the images, regardless of multi-scale or dense regions, even under turbid water and lighting conditions, reducing the missed detection rate. Overall, the ResNetCBAM-FUSSION-YOLO network can rapidly, accurately, and comprehensively detect fish schools in harsh environments. This is of significant importance for understanding the growth status of aquaculture fish and achieving precise feeding.

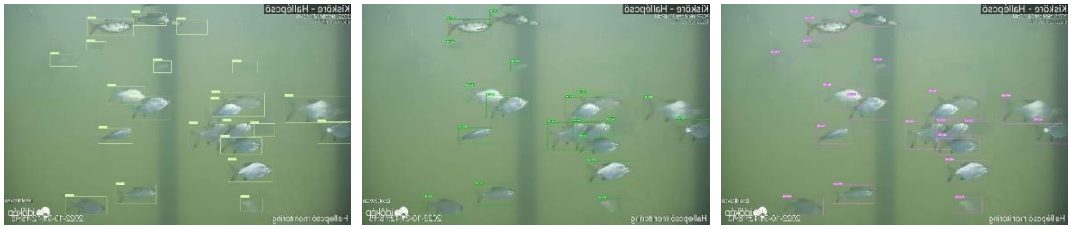

Figure 11. ResNetCBAM-FUSSION-YOLO

Figure 12. YOLOV7

Figure 13. YOLOV7-Tiny

4. Discussion

The population detection model for sea bream proposed in this paper ensures accuracy based on lightweighting, but there are still some key issues that need further exploration. Firstly, the dataset plays a crucial role in the target detection model. Currently, we only focus on detecting sea bream underwater, which still poses a risk of model performance degradation for other fish species. Additionally, during the dataset annotation process, due to the large workload and subjective nature, errors in labeling are inevitable. In future research, we consider using computer-assisted human labeling or employing pre-trained models and manually modifying the outputs of pre-trained models to ensure optimal labeling effectiveness.

Furthermore, when deploying the model, hardware resources need to be considered. Although we used FasterNet, which significantly reduced the model's parameter count and computational load, other methods can still be employed for lightweighting the model. For instance, reducing the model size by removing redundant connections or unnecessary parameters, or employing knowledge distillation where the output of a teacher model is used as auxiliary information to train a student model. These methods can enhance lightweighting efficiency and stability.

Lastly, the data augmentation techniques employed in the experiments to expand the dataset may not fully consider all variations in complex aquaculture environments. Therefore, further research is needed to enhance the model's adaptability to different environments and scenarios, thereby improving the feasibility and stability of practical applications.

5. Conclusion

(1) By collecting camera data and manually labeling images using Labelimg software, a dataset for detecting sea bream in complex environments was created. In order to enhance the robustness of the model, data augmentation techniques were also applied during the dataset creation process.

(2) The FasterNet backbone was used to replace the original YOLOv7-TINY algorithm, aiming to reduce the model's parameter count and computational complexity, as well as accelerate detection speed, while optimizing performance without significantly compromising accuracy. This lays the foundation for subsequent network optimization.

(3) Building upon (2), the three layers provided by the Backbone were utilized as inputs to the Fusion module to further extract contextual information from the model. Before feeding into the detection head, the RESNETCBAM attention mechanism was applied to features of three different scales, enhancing the model's ability to integrate multi-scale information and texture features. To accelerate training speed, the EIOU and CIOU loss functions were combined to form ECIOU, which not only enhances the accuracy, robustness, and stability of the bounding box matching metric in object detection tasks but also incorporates the advantages of both loss functions.zation.

In future work, we will further expand the types of fish

populations that need to be detected in aquaculture and different scenarios of aquaculture environments. We aim to establish a digital fishery system by integrating sensors such as dissolved oxygen and pH meters, to achieve more accurate and comprehensive assessment of fish growth conditions. The goal is to improve fisheries production efficiency and optimize resource utilization. Additionally, we will enhance the corresponding detection algorithms to find a better balance between speed and accuracy.

Overall, the proposed sea bream group object detection model demonstrates high efficiency and is suitable for deployment on mobile devices, providing strong technical support for aquaculture technology.

Acknowledgment

Funding: This work was supported in part by the National Natural Science Foundation of China un-der Grant 62175037, in part by the Huzhou Key R&D Program Agricultural “Double Strong” Special Project (No. 2022ZD2060),in part by Zhejiang-French Digital Monitoring Lab for Aquatic Resources and Environment, Department of Science and Technology of Zhejiang Province, and in part by the Huzhou Key Laboratory of Waters Robotics Technology (2022-3), Huzhou Science and Technology Bureau.

References

[1] Xu Hai, Xie Hongtao, and Zhang Yongdong, "Advances in Visual Domain Generalization Techniques and Research," Journal of Guangzhou University (Natural Science Edition), vol. 21, no. 2, pp. 42–59, 2022.

[2] N. J. C. Strachan, P. Nesvadba, and A. R. Allen, "Fish species recognition by shape analysis of images," Pattern Recognition, vol. 23, no. 5, pp. 539–544, January 1990, doi: 10.1016/0031-3203(90)90074-U.

[3] N. Castignolles, M. Cattoen, and M. Larinier, "Identification and counting of live fish by image analysis," presented at IS&T/SPIE 1994 International Symposium on Electronic Imaging: Science and Technology, S. A. Rajala and R. L. Stevenson, eds., San Jose, CA, March 1994, pp. 200–209. doi: 10.1117/12.171067.

[4] D.-J. Lee, R. B. Schoenberger, D. Shiozawa, X. Xu, and P. Zhan, "Contour matching for a fish recognition and migration-monitoring system," presented at Optics East, K. G. Harding, ed., Philadelphia, PA, December 2004, p. 37. doi: 10.1117/12.571789.

[5] Ding Shunrong and Xiao Ke, "Research on fish classification method based on particle swarm optimization SVM and multi-feature fusion," Chinese Journal of Agricultural Mechanization, vol. 41, no. 11, pp. 113–118, 170, 2020, doi: 10.13733/j.jcam.issn.2095-5553.2020.11.018.

[6] Yao Runlu, Gui Yongwen, and Huang Qiugui, "Freshwater fish species recognition based on machine vision," Journal of Microcomputers and Applications, vol. 36, no. 24, pp. 37–39, 2017, doi: 10.19358/j.issn.1674-7720.2017.24.011.

[7] P. Cisar, D. Bekkozhayeva, O. Movchan, M. Saberioon, and R. Schraml, "Computer vision based individual fish identification using skin dot pattern," Sci Rep, vol. 11, no. 1, p. 16904, August 2021, doi: 10.1038/s41598-021-96476-4.

[8] R. B. Dala-Corte, J. B. Moschetta, and F. G. Becker, "Photo-identification as a technique for recognition of individual fish: a test with the freshwater armored catfish Rineloricaria aequalicuspis Reis & Cardoso, 2001 (Siluriformes: Loricariidae)," Neotrop. ichthyol., vol. 14, no. 1, 2016, doi: 10.1590/1982-0224-20150074.

[9] Chen Feifen, "Research and application of water meter reading recognition based on deep learning," Master's thesis, Guilin University of Electronic Technology, 2022. doi: 10.27049/d.cnki.ggldc.2021.000020.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[11] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv, April 10, 2015. Accessed: March 14, 2024. [Online]. Available: http://arxiv.org/abs/1409.1556

[12] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," arXiv, February 19, 2015. Accessed: March 14, 2024. [Online]. Available: http://arxiv.org/abs/1409.2329

[13] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," IEEE Trans. Neural Netw. Learning Syst., vol. 28, no. 10, pp. 2222–2232, October 2017, doi: 10.1109/TNNLS.2016.2582924.

[14] I. Goodfellow et al., "Generative Adversarial Nets."

[15] C. Szegedy et al., "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA: IEEE, June 2015, pp. 1–9. doi: 10.1109/CVPR.2015.7298594.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, June 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[17] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv, April 16, 2017. Accessed: March 14, 2024. [Online]. Available: http://arxiv.org/abs/1704.04861

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT: IEEE, June 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.

[19] A. Howard et al., "Searching for MobileNetV3," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South): IEEE, October 2019, pp. 1314–1324. doi: 10.1109/ICCV.2019.00140.

[20] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in 2017

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation."

[22] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, June 2017, doi: 10.1109/TPAMI.2016.2577031.

[23] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," presented at Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2961–2969. Accessed: March 14,2024. [Online]. Available: https: //openaccess.thecvf.com/content_iccv_2017/html/He_Mask_ R-CNN_ICCV_2017_paper.html

[24] W. Liu et al., "SSD: Single Shot MultiBox Detector," in Computer Vision – ECCV 2016, vol. 9905, B. Leibe, J. Matas, N. Sebe, and M. Welling, eds., Lecture Notes in Computer Science, vol. 9905., Cham: Springer International Publishing, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

[25] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv, April 8, 2018. doi: 10.48550/arXiv.1804.02767.

[26] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv, April 22, 2020. doi: 10.48550/arXiv.2004.10934.

[27] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios," in 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, BC, Canada: IEEE, October 2021, pp. 2778–2788. doi: 10.1109/ICCVW54120.2021.00312.

[28] C. Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications," arXiv, September 7, 2022. doi: 10.48550/arXiv.2209.02976.

[29] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada: IEEE, June 2023, pp. 7464–7475. doi: 10.1109/CVPR52729.2023.00721.

[30] S. Zhao, J. Zheng, S. Sun, and L. Zhang, "An Improved YOLO Algorithm for Fast and Accurate Underwater Object Detection," Symmetry, vol. 14, no. 8, Art. no. 8, August 2022, doi: 10.3390/sym14081669.

[31] Y. Li, X. Bai, and C. Xia, "An Improved YOLOV5 Based on Triplet Attention and Prediction Head Optimization for Marine Organism Detection on Underwater Mobile Platforms," JMSE, vol. 10, no. 9, p. 1230, September 2022, doi: 10.3390/jmse10091230.

[32] X. Zhai, H. Wei, Y. He, Y. Shang, and C. Liu, "Underwater Sea Cucumber Identification Based on Improved YOLOv5," Applied Sciences, vol. 12, no. 18, p. 9105, September 2022, doi: 10.3390/app12189105.

[33] A. Markus, G. Kecskemeti, and A. Kertesz, "Flexible Representation of IoT Sensors for Cloud Simulators," in 2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), March 2017, pp. 199–203. doi: 10.1109/PDP.2017.87.

[34] J. Chen et al., "Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks," arXiv.org .Accessed:November 8, 2023. [Online]. Available: https://arxiv.org/abs/2303.03667v3

[35] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," presented at Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856. Accessed: March 15, 2024. [Online] .Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_ShuffleNet_An_Extremely_CVPR_2018_paper.html

[36] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More Features From Cheap Operations," presented at Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1580–1589. Accessed: March 15, 2024. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2020/html/Han_GhostNet_More_Features_From_Cheap_Operations_CVPR_2020_paper.html

[37] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks," in Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, June 2011, pp. 315–323. Accessed: March 15, 2024. [Online]. Available: https://proceedings.mlr.press/v15/glorot11a.html

[38] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," presented at Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3-19. Accessed: March 15, 2024 .[Online]. Available:https://openaccess.thecvf.com/content_ECCV_2018/html/Sanghyun_Woo_Convolutional_Block_Attention_ECCV_2018_paper.html

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, pp. 533–536, October 1986, doi: 10.1038/323533a0.

[40] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression," presented at Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp.658–666.Accessed: March 15,2024.[Online].Available:https://openaccess.thecvf.com/content_CVPR_2019/html/Rezatofighi_Generalized_Intersection_Over_Union_A_Metric_and_a_Loss_for_CVPR_2019_paper.html

[41] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 07, Art. no. 07, April 2020, doi: 10.1609/aaai. v34i07.6999.

[42] Z. Zheng et al., "Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation," IEEE Transactions on Cybernetics, vol. 52, no. 8, pp. 8574–8586, August 2022, doi: 10.1109 / TCYB.2021.3095305.