Received: 28 September 2025 | Revised: 27 March 2026 | Accepted: 24 April 2026 | Published online: 21 May 2026

RESEARCH ARTICLE

Journal of Data Science and Intelligent Systems
2026, Vol. 00(00) 1–13
DOI: 10.47852/bonviewJDSIS62027779

BON VIEW PUBLISHING

Underwater Seafood Detection Using Deep Learning and Data Augmentation

Hanyu Jiang[1, 2] **and Zhichao Zheng**[3,*]

[1] *State Key Laboratory of Robotics and Intelligent Systems, Chinese Academy of Sciences, China*

[2] *University of Chinese Academy of Sciences, China*

[3] *School of Electrical Engineering and Automation, Wuhan University, China*

Abstract: Accurate seafood detection underwater is still a real challenge for modern fisheries, especially when it comes to resource monitoring or automated harvesting. In real-world conditions, underwater images are often blurry, discolored, or cluttered with background noise, all of which make feature extraction harder. On top of that, many marine species have irregular, elongated body shapes, which adds another layer of difficulty for reliable detection. To tackle these problems, we developed an underwater seafood detection model called Seafood Detection Deep Learning (SDDL). It builds on the YOLOv8 framework but integrates dynamic snake convolution to better handle elongated features like sea urchin spines or sea cucumber tentacles. This change lets the network deal more effectively with anisotropic shapes. Before training, we apply Contrast Limited Adaptive Histogram Equalization to boost image contrast and bring out finer details. We also use Focal Loss during training to reduce the impact of class imbalance. The dataset we used contains 5543 underwater images, which provides a realistic and fairly challenging testbed. SDDL achieves 84.22% mAP50 and 48.74% mAP50–95 on this dataset. When we compared it against several representative methods, SDDL gave more consistent results under difficult imaging conditions. Overall, these findings suggest the model holds good promise for deployment in underwater robotic systems, helping enable accurate seafood detection and supporting intelligent, automated harvesting in fisheries.

Keywords: underwater image, image enhancement, dynamic snake convolution, CLAHE, YOLOv8

1. Introduction

Seafood is one of the most widely consumed categories of marine products around the world [1–3], playing a significant role in both human nutrition and global trade. That said, telling different seafood species apart is often far from easy. The sheer variety of products, combined with complex shapes and close visual similarities between some types, makes accurate identification a real challenge in practice.

In many real-world settings, identification still depends heavily on manual inspection and expert knowledge [4–7]. These approaches tend to be time-consuming and prone to subjective judgment. As a result, they introduce an inherent source of inconsistency, which makes it hard to meet the high standards of efficiency and accuracy that modern food safety inspection demands.

Developing a detection method for hidden seafood that is efficient, accurate, and automated has therefore become an important research objective [8]. With the rapid development of computer vision technology, image recognition has gradually been introduced as an effective tool for seafood identification. Among these techniques, deep learning [9] has shown strong capability in visual analysis and pattern recognition. As a core architecture in deep learning, convolutional neural networks (CNNs) have been extensively adopted for image-related tasks because of their strong feature extraction ability and stable performance in various recognition scenarios [10–12].

Object detectors have achieved strong performance in many vision tasks [13–15], including several underwater applications [16–19]. However, underwater object detection remains a challenging problem. Identifying seafood species is particularly difficult because many species share similar visual appearances, and their features are often affected by environmental conditions. As a result, detection models need not only effective feature extraction capability but also the ability to adapt to complex underwater environments. Compared with natural images, underwater images are strongly influenced by light attenuation and scattering, which often causes color distortion, reduced contrast, and image blur. These factors significantly increase the difficulty of recognizing underwater targets. To alleviate these problems, many existing studies attempt to improve general object detection frameworks by incorporating dedicated modules or adaptation strategies designed for underwater scenes. Several representative approaches have been proposed. FERNet [20] adopts a dual-refinement framework consisting of semantic and positional refinement branches to improve detection performance.

***Corresponding author:** Zhichao Zheng, School of Electrical Engineering and Automation, Wuhan University, China. Email: zzc0101@whu.edu.cn

SWIPENet [21] applies a multi-class AdaBoost training strategy together with a sample-weighted loss, enabling the backbone network to generate high-resolution feature maps with richer semantic information, which benefits small-object detection in noisy underwater environments. Boosting R-CNN [22] redesigns the RetinaRPN to generate region proposals and combines probabilistic inference with hard-sample mining to improve robustness on underwater datasets. ERL-Net [23] introduces an edge-guided representation learning framework that integrates edge information with multilevel features, improving detection performance in low-contrast and small-object scenarios. Recent work has also explored the integration of attention mechanisms into underwater detection frameworks to strengthen feature representation [24, 25]. For example, Liang and Song [26] improve region-of-interest (RoI) features through an external attention module to refine detection results. Another study by Shen et al. [27] proposes a multidimensional and multilevel attention block that combines different attention strategies to improve robustness in complex underwater backgrounds. In this work, dynamic snake convolution (DSC) and Focal Loss are adopted to address the elongated structural characteristics of seafood targets and the class imbalance problem in underwater datasets, enabling more effective feature extraction for seafood recognition.

Another set of work has looked at improving things from the data side, mainly through better preprocessing and augmentation to boost model performance. The idea is to help models learn stronger features and become more robust under murky underwater conditions. This addresses a real limitation caused by poor image quality. For instance, RoIMix [28] proposes a region-level augmentation method that mixes several RoI regions within one image, creating overlapping and occluded scenarios. That forces the detector to learn more general object representations for underwater tasks. The added feature diversity works like a kind of implicit regularization during training. Similarly, Poisson GAN and AquaNet [29] use multi-branch architectures to combine local region cues with semantic information. Synthetic images generated by Poisson GAN further increase training diversity and improve detection. The fusion of different types of features creates an anisotropic representation space that captures underwater variability better. Some studies have focused on underwater image enhancement as a preprocessing step to reduce degradation before detection. One representative method [30] introduces an object-guided dual adversarial contrastive learning framework that helps correct distortion in underwater images. That leads to noticeable improvements in visual quality that directly benefit downstream detection. Another approach by Fu et al. [31] proposes a Residual Feature Transference Module, which extracts transferable priors from enhanced images and uses them to refine degraded feature distributions. This mechanism strengthens detection in visually tough environments, offering a subtle but effective alignment of features across domains. In our own work, we apply Contrast Limited Adaptive Histogram Equalization (CLAHE) during preprocessing to boost contrast and improve image clarity. That gives us more distinguishable visual patterns for the detection stages that follow, leading to a fine-grained improvement in recognition accuracy.

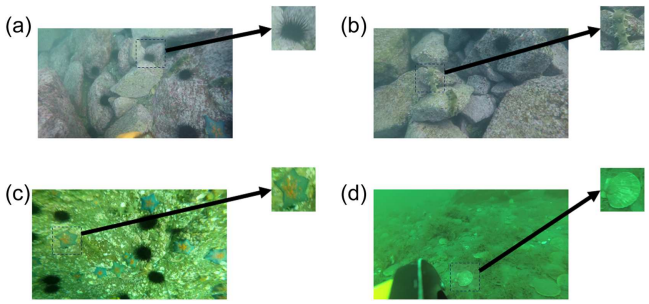
This work presents a seafood detection framework based on YOLOv8, with a few data augmentation tricks to make it work better underwater. Underwater images are often distorted or blurry, which messes with reliable feature extraction. So we apply CLAHE during preprocessing to clean up the visuals a bit—a small step that tackles a real pain point caused by poor image quality. We also throw in DSC to help the model handle those long, skinny shapes you often see in marine creatures. On top of that, we use Focal Loss during training to deal with class imbalance in the dataset. Put together, these pieces form a fairly unique optimization setup for underwater detection. The final model can be deployed on underwater robots for seafood identification. With this framework, robotic arms can carry out automated harvesting more precisely. That integration actually contributes something meaningful to intelligent underwater operations and pushes unmanned seafood harvesting a step forward. To sum up, the main contributions of this paper are:

1) We propose a model that makes good use of data augmentation. Thanks to the augmented data, our method achieves higher accuracy.
2) Extensive experiments show that our model actually works.
3) In side-by-side comparisons, our model beats several other methods and delivers the best performance.

2. Methodology

The dataset used in this study was captured from a marine ranch on Zhangzi Island in Dalian, China. It comprises a total of 5543 underwater images. There are four target categories (Figure 1) in this dataset: sea urchins, sea cucumbers, starfish, and scallops. The dataset is divided into three parts: the training set contains 3880 images, the validation set contains 554 images, and the test set contains 1109 images. All images are sourced from the public URPC dataset.

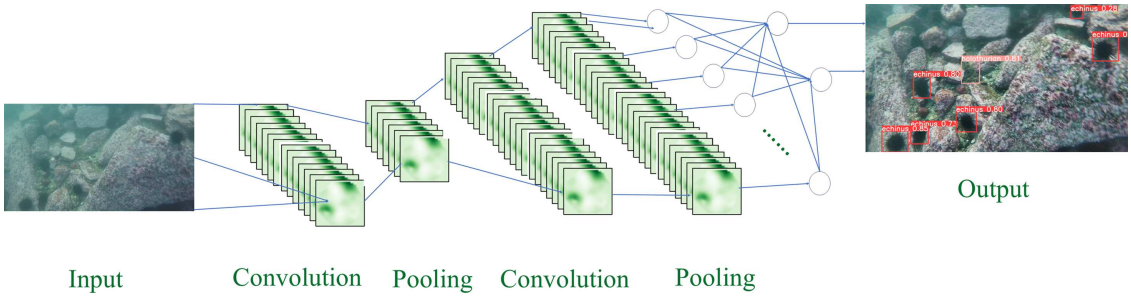
Figure 1
Images of different categories in the dataset: (a) sea urchins, (b) sea cucumbers, (c) starfish, and (d) scallops



2.1. Neural network

Object detection networks are a class of deep learning models designed to figure out where objects are in an image or video and what they are. In our case, the input is underwater images showing four types of seafood: sea urchins, sea cucumbers, starfish, and scallops. The model takes these images and produces annotated outputs with detected objects boxed and labeled, as shown in Figure 2. That's basically the whole visual recognition pipeline. A typical object detection framework has a few standard building blocks: convolutional layers, pooling layers, fully connected layers, plus some specialized modules for classification and localization. Convolutional layers use multiple learnable filters that slide over the image, do a bit of math, add a bias, and then pass the result through a nonlinear activation. This produces feature maps that capture local patterns unique to each image. As you go deeper into the network, these layers start pulling

Figure 2
Schematic illustration of the object detection neural network

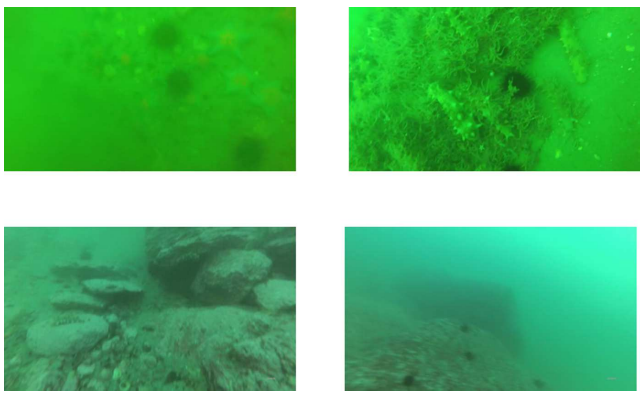


out higher-level semantic information—exactly what you need for identifying underwater targets. Pooling layers then downsample the feature maps spatially, compressing them while trying to keep the important structural information intact. This step reduces resolution unevenly but retains the meaningful content. After several rounds of feature extraction, the network builds up a decent representation of the scene. Finally, the fully connected layers take over, pulling everything together to decide what objects are present and where their bounding boxes should go. All of this works together as a fairly integrated system for making accurate detections.

2.2. Data augmentation

Underwater imaging conditions differ significantly from those encountered in natural scenes. In aquatic environments, light is gradually attenuated and scattered as it propagates through water, which often results in blurred images and visual distortion. These effects increase the difficulty of accurately identifying underwater targets. In addition, the absorption and scattering characteristics of light in water are different from those in air, frequently causing noticeable color distortion that can reduce the visibility of objects and influence recognition performance. The underwater environment also contains bubbles, suspended particles, and sediments that introduce noise and interfere with image quality. Furthermore, many underwater organisms exhibit irregular shapes and complex surface textures that differ from typical land-based objects. Such characteristics make reliable feature extraction more difficult, especially for detection approaches that rely heavily on texture information. Examples of underwater images are presented in Figure 3.

Figure 3
Examples of underwater images



Conventional Adaptive Histogram Equalization (AHE) [32] often amplifies contrast in regions of the image that are nearly constant, due to the high concentration of histogram values in such areas. As a result, AHE may amplify noise in these nearly constant regions. CLAHE [33] is an improved version of AHE that constrains contrast enhancement to prevent excessive noise amplification. In CLAHE, the degree of contrast enhancement around a pixel is governed by the gradient of the transformation function, which is related to the Cumulative Distribution Function (CDF) and thus reflects the local histogram distribution. For a given image x, the CDF formula is shown in the equation:

$$cdf_x(i) = \sum_{j=0}^{i} p_x(x = j) \tag{1}$$

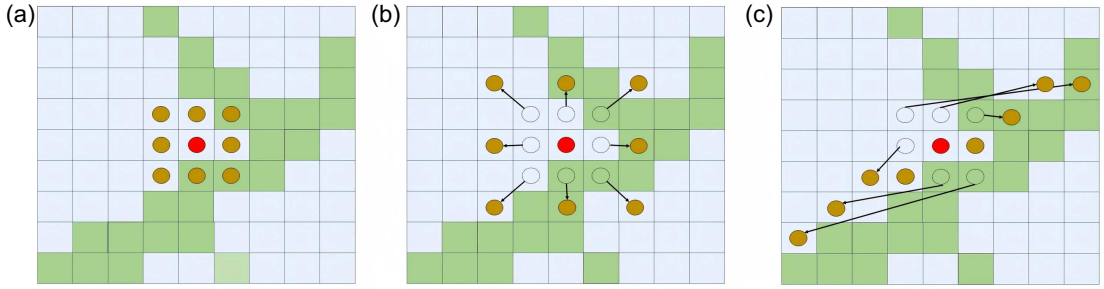
where i and j both represent a particular grayscale value and P denotes probability.

CLAHE restricts contrast enhancement by first clipping the histogram prior to computing the CDF, which in turn limits the CDF's slope and consequently the slope of the transformation function. The clipping limit, which determines how much of the histogram is clipped, depends on the histogram's normalization and hence on the neighborhood size. Typically, the limit values constrain the amplification factor to between 3 and 4. Importantly, instead of discarding histogram portions beyond the clipping limit, they are evenly redistributed across all histogram bins. In our work, we applied CLAHE for data augmentation with impressive results: underwater images became clearer, aiding deep learning models in learning crucial features.

2.3. Dynamic snake convolution

Standard convolution [34] (see Figure 4(a)) is a basic building block in computer vision and deep learning, particularly in CNNs. Its main job is to pull out representative features from input data by processing local regions. That localized operation helps the network pick up meaningful spatial patterns, which is a pretty fundamental step for visual understanding. However, each convolutional layer only sees a small receptive field. That naturally limits how well the network can capture long-range dependencies or global context. When you're dealing with complex spatial layouts or heavy background clutter, this limitation can hurt detection accuracy. It's basically a built-in weakness when it comes to modeling broader interactions. One common fix is to stack several convolutional layers, so information can gradually travel across a larger area. Deeper networks do cover more ground, but they also become more complex to train. That increased depth gives you better representational power, but it also comes with anisotropic growth in difficulty—things like

Figure 4
(a) Standard convolution, (b) dilated convolution, and (c) dynamic snake convolution



vanishing or exploding gradients become real issues. On top of that, standard convolution uses fixed-size kernels, which makes it less flexible when dealing with features at different scales or distances. So in diverse visual conditions, this rigidity turns into a real bottleneck for flexible feature modeling.

Dilated convolution [35] (see Figure 4(b)) works by sticking gaps between adjacent entries in the convolutional kernel, which effectively expands the receptive field. This lets the network see a larger spatial area without having to add more parameters or go deeper. It's a neat way to improve contextual awareness. The nice thing is that it keeps the output feature map at the original resolution while pulling in broader contextual information. That makes it handy for tasks that need both a large receptive field and fine spatial detail—think semantic segmentation, object detection, or image super-resolution. It strikes a pretty interesting balance between keeping resolution and gathering context. That said, if you make the dilation rate too large, the convolution starts skipping over nearby pixels and loses sensitivity to local structures. In that case, the model leans too heavily on global context and may miss fine-grained details. This ends up hurting local feature representation unevenly, which can become a real problem when you're trying to detect small or densely packed objects.

DSC [36] (see Figure 4(c)) offers a more flexible way of doing convolution by letting the kernel shape and its sampling path adjust to the geometric patterns in the input image. Standard convolution sticks to fixed kernels and regular sampling grids. DSC, on the other hand, changes its sampling route based on local structural cues. That makes the whole feature extraction process much more adaptive. The sampling path tends to follow irregular or curved shapes, so the operation aligns better with the actual geometry of the object. This adaptability is what makes DSC good at handling elongated or complex morphological features [15, 36]. In practice, the mechanism is quite sensitive to structural variations in the feature maps. Another nice thing is that it keeps computational cost relatively low while focusing on the informative parts of the feature space. That focus helps bring out subtle signals that standard convolution often misses. The result is a more refined, anisotropic adjustment of feature responses. Inside the DSC operation, the convolution kernel shifts dynamically along both the x- and y-directions, as shown in Figure 4(c). The movement along the x-direction follows a specific formulation, which we give below. This formulation provides a clear description of how the kernel adapts on the fly.

$$K_{i\pm c} = \begin{cases} (x_{i+c}, y_{i+c}) = \left(x_i + c, y_i + \sum_i^{i+c} \Delta y\right) \\ \\ (x_{i-c}, y_{i-c}) = \left(x_i - c, y_i + \sum_{i-c}^{i} \Delta y\right) \end{cases} \tag{2}$$

For the y-direction, the convolution kernel follows the formulation below. This just takes the adaptive sampling mechanism one step further.

$$K_{j+c} = \begin{cases} (x_{j+c}, y_{j+c}) = \left(x_j + \sum_j^{j+c} \Delta x, y_j + c\right) \\ \\ (x_{j-c}, y_{j-c}) = \left(x_i + \sum_{j-c}^{j} \Delta x, y_i - c\right) \end{cases} \tag{3}$$

In this formulation, $K_{(i\pm c)} = (x_j, y_j)$ gives the center coordinate of a standard 2D convolution kernel. The terms Δx and Δy are the deformation offsets along the horizontal and vertical directions. This way of parameterizing things introduces a decent amount of spatial flexibility. Because the offsets can adapt, the kernel's sampling points shift according to the local structure in the image. Earlier work has shown that this kind of mechanism works particularly well for elongated or tube-like shapes. It's basically a built-in advantage when dealing with irregular geometries. When you plug this into a deep learning architecture, it strengthens the extraction of complex geometric features and leads to better detection results. That turns out to be really useful in underwater environments, where many organisms have irregular or stretched-out shapes. In effect, you get an anisotropic boost in feature representation. Integrating DSC into underwater seafood detection helps the model do a better job with slender structures and subtle morphological differences. The resulting features support more accurate recognition of target objects. Overall, this improvement makes a clear contribution to detection accuracy in challenging scenarios.

2.4. Elongated features in the dataset

DSC was designed to identify fine, elongated structures in specific domains. Concurrently, we discovered that our task also involves fine elongated structures that can be efficiently captured by DSC. In Figure 1(a), sea urchins are covered with fine, elongated spines. In Figure 1(b), sea cucumbers exhibit a generally elongated form, including their tentacles. In Figure 1(c), we observe that the arms of the starfish possess elongated features, although less pronounced compared to other species, which might result in a relatively smaller contribution of DSC to starfish compared to sea cucumbers. In Figure 1(d), scallops, while lacking prominent overall elongation, feature fine, elongated striations, a critical characteristic that DSC focuses on learning and capturing.

2.5. The loss function in object detection

Intersection over Union (IoU) [37] is a pretty standard metric in object detection. It basically tells you how well a predicted

bounding box matches up with the ground-truth annotation. If the IoU is high, that means the two boxes overlap a lot, and the prediction is accurate. If it's low, the alignment is off. This gives you a solid basis for judging localization quality. Formally, IoU is defined as the area where the predicted box and the ground-truth box overlap (Figure 5(a)) divided by the total area covered by both boxes combined (Figure 5(b)). That ratio captures both how much they agree and how much they disagree in one simple number. It's a neat way to look at spatial correspondence. The calculation of IoU follows directly from this ratio. The resulting number gives you a quantitative measure of detection accuracy. In practice, this metric is quite sensitive to how well the predicted region aligns with the reference region.

$$IOU = \frac{Area_{Intersection}}{Area_{Union}} \tag{4}$$

Complete Intersection over Union (CIoU) [38] is a pretty common loss function for bounding box regression in object detection. Compared to the traditional IoU-based loss, CIoU takes into account additional geometric relationships between the predicted box and the ground-truth box. That extra bit of information makes it a clear step up in terms of modeling spatial constraints. Specifically, the loss doesn't just look at how much the two boxes overlap. It also considers the distance between their center points, as well as differences in scale and aspect ratio. Pulling all of that together gives you a more complete picture of geometric alignment. During training, these extra factors provide richer guidance, helping the model get a better handle on both where an object is and what shape it roughly takes. The end result is more precise localization. In effect, the loss becomes sensitive to both positional mismatches and structural differences. The mathematical expression of the CIoU loss is given below.

$$\mathscr{L}_{CIoU} = 1 - IoU + \frac{\rho^2(\mathrm{b}, \mathrm{b}^{gt})}{c^2} + \alpha v \tag{5}$$

In this expression, ρ^2 represents the squared distance between the centers of the predicted and ground-truth bounding boxes. The term c is treated as a constant, usually defined by the diagonal span of the enclosing region formed by both boxes. This formulation introduces a nontrivial geometric constraint. The parameter α acts as a weighting factor that adjusts the contribution of different components within the loss. The variable v is included as a penalty term that captures discrepancies in aspect ratio between the predicted and ground-truth boxes. Such a design reflects an idiosyncratic treatment of shape inconsistency.

In this expression, ρ^2 is the squared distance between the centers of the predicted box and the ground-truth box. The term c is treated as a constant—typically the diagonal length of the smallest box that encloses both the prediction and the ground truth. This setup adds a meaningful geometric constraint to the loss. The parameter α is a weighting factor that adjusts how much each part of the loss contributes. Then there's v, which acts as a penalty term for differences in aspect ratio between the predicted box and the ground-truth box. This whole design handles shape mismatches in a fairly distinctive way.

Enhanced Intersection over Union (EIOU) [39] builds on the standard IoU metric by offering a more refined loss function for bounding box regression in object detection. It's essentially an upgrade to CIoU, adding a few extra considerations to address some known shortcomings. That makes it a decent step forward in loss function design. The formulation is more adaptable to objects of different shapes and also speeds up bounding box regression. On top of that, it handles the imbalance among training samples better by providing smoother optimization behavior. Taken together, these improvements give the model a richer learning signal, which translates into better localization accuracy. As a result, the optimization process produces more reliable bounding box predictions across a variety of conditions. In effect, the loss becomes sensitive to both spatial and structural differences in a more nuanced way. The mathematical formulation of EIOU is given below.

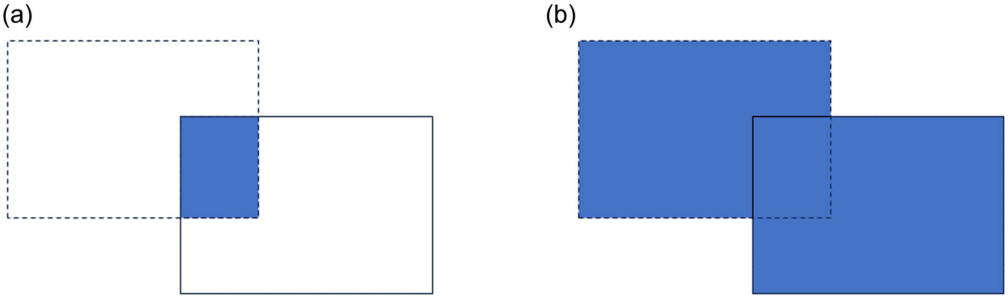
$$\begin{aligned} L_{EIOU} &= L_{IOU} + L_{dis} + L_{asp} \\ &= 1 - IOU + \frac{\rho^2(\mathrm{b}, \mathrm{b}^{gt})}{(w^c)^2 + (h^c)^2} + \frac{\rho^2(w, w^{gt})}{(w^c)^2} + \frac{\rho^2(h, h^{gt})}{(h^c)^2} \end{aligned} \tag{6}$$

The terms h^c and w^c represent the height and width of the smallest bounding box that covers both. This serves as a useful geometric anchor.

Focal Loss [40] is a tweaked version of cross-entropy loss, often used to deal with class imbalance in training data. In standard cross-entropy, every misclassified sample gets the same treatment, which can bias the model toward the majority classes when the dataset is skewed. That poses a real challenge during optimization. Focal Loss adds a modulating factor that tones down the influence of easy-to-classify samples. Instead, it puts more weight on the hard ones, pushing the model to learn more from those difficult examples. This weighting scheme essentially adjusts how important each sample is during training. Samples with predicted probabilities close to 0 or 1 get lower weights, while those with probabilities somewhere in the middle get higher weights. That way, the model's learning focus gets distributed

Figure 5

Illustration of IoU in an image: the solid rectangle denotes the ground-truth bounding box, while the dashed rectangle indicates the predicted bounding box: (a) the overlapping region between the two boxes and (b) the total area enclosed by both, providing a nontrivial visualization of spatial correspondence



unevenly across samples, paying more attention to the ones it's still uncertain about. The mathematical formulation of Focal Loss is given in the equation below.

$$\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{7}$$

In this formulation, p_t is the probability the model assigns to the correct class. The coefficient α_t acts as a balancing weight across different categories—it helps adjust for class distribution. That's a useful tweak when your dataset is uneven. The parameter γ is a focusing factor that controls how much emphasis is placed on hard-to-classify samples. If you set γ larger, the model pays more attention to those difficult examples during training. So essentially, this mechanism gives you a way to prioritize certain samples over others.

3. Proposed Methods

3.1. The SDDL algorithm proposed in this paper

After gathering a substantial number of underwater seafood images, the research team conducted detailed manual annotations in the laboratory, marking the exact locations and categories of each seafood instance. The annotated dataset was then systematically partitioned into training, validation, and testing subsets, as depicted in Figure 6(a). Before training, all images are enhanced using CLAHE to improve their perceptibility and details, which facilitates the subsequent training and helps achieve higher accuracy. These subsets were subsequently input into the Seafood Detection Deep Learning (SDDL) framework for model training, evaluation, and performance assessment. The internal configuration of the SDDL module is presented in Figure 6(b). This module receives images captured by visual sensors and processes them to generate prediction results, which are used for object detection tasks [41–46]. In Figure 6(c), two loss functions are depicted: Bbox Loss and Cls Loss. These correspond to bounding box regression and category classification tasks, respectively. By optimizing these two loss functions, we can obtain a deep learning model capable of accurately identifying target positions and correctly classifying target categories.

The architectural design of the SDDL network is illustrated in Figure 7, while Figure 7(a) outlines the overall detection workflow of the model.

Step 1: Underwater images first go through the backbone network, which has DSC built into it. DSC helps the model respond adaptively to local geometric patterns in underwater targets, making feature extraction and spatial alignment work better. This step gives a solid boost to structural representation.

Step 2: The feature maps from the backbone are then passed to the Head module, where concatenation and upsampling operations merge multi-scale information. This lets the network keep fine details while also capturing higher-level contextual cues. It's a fairly clever way to aggregate features across different scales.

Step 3: After the Backbone and Head stages, the feature maps carry richer spatial and semantic information. They're then fed into the Detect module for object prediction. This transition creates a directional flow of information from feature extraction to localization.

Inside the Detect module, the feature maps are decoded, and anchor boxes are generated for each grid cell following a predefined scheme. The overlap between anchors and ground-truth boxes is then computed to match anchors with actual targets—a key step for estimating correspondences. The predictions include category probabilities and bounding box adjustments for each

Figure 6
(a) Workflow of SDDL network and (b) internal structure of SDDL network

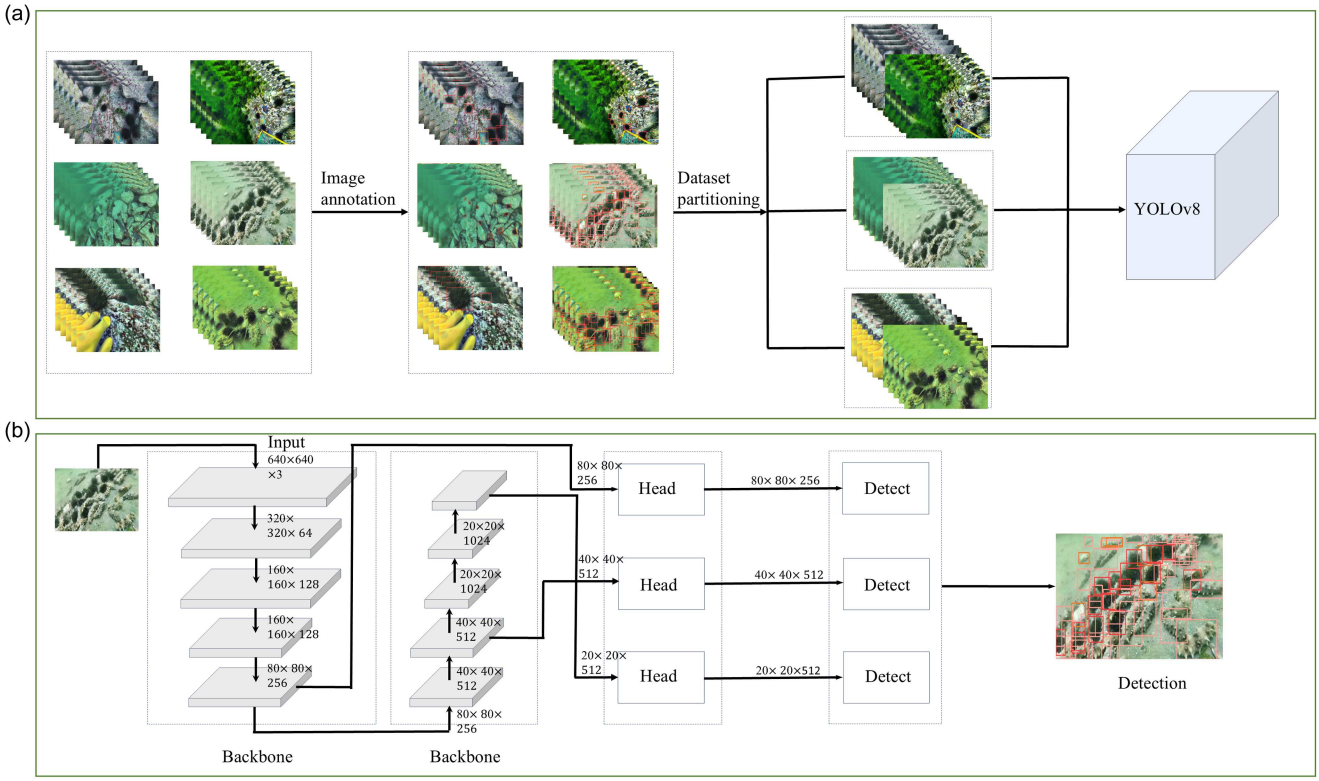
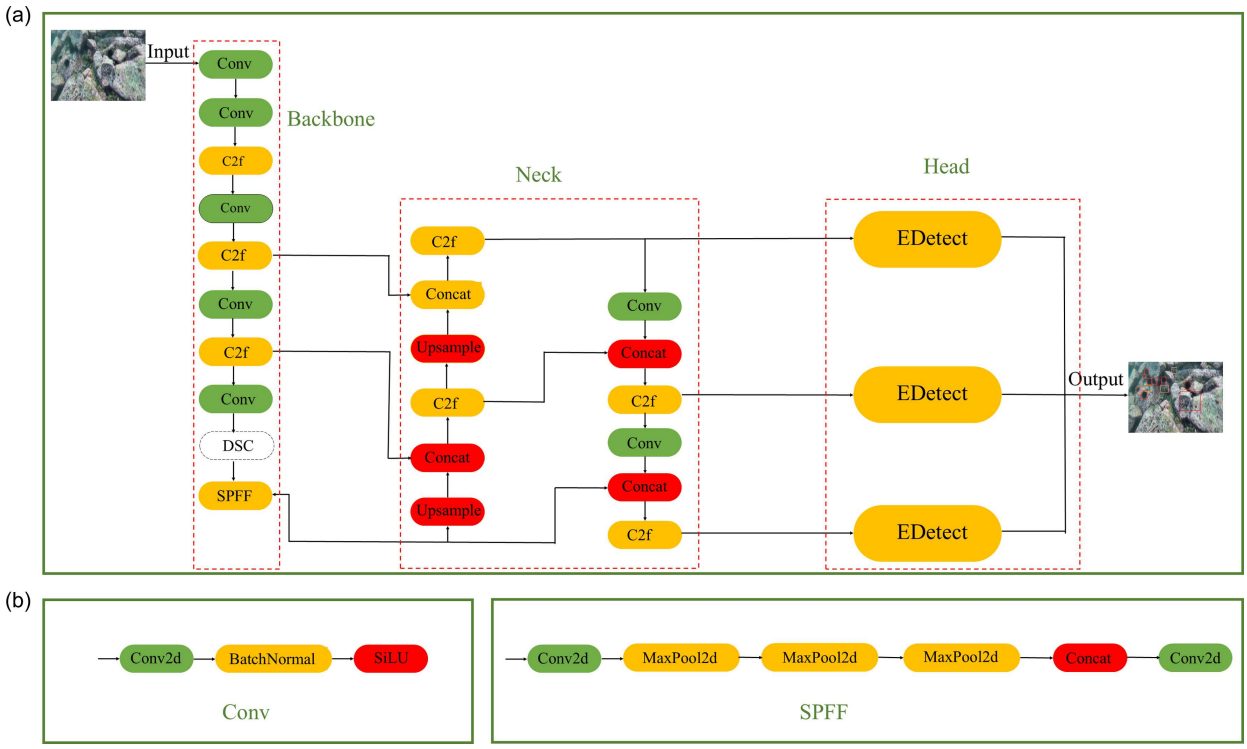


Figure 7
(a) Overall architecture of SDDL network and (b) detailed internal structure of some modules



anchor. The probabilities tell you how likely each category is, while the bounding box offsets fine-tune the position and scale to better fit the target objects. Together, these give a fairly detailed picture of the detection outputs. Through these steps, the network manages to localize and classify seafood targets effectively in underwater images. The structures of several key modules are shown in Figure 7(b). This setup provides a step-by-step pipeline for feature processing and prediction. In our proposed framework, we apply CLAHE during preprocessing to boost image contrast and improve visual clarity. DSC is embedded in the YOLOv8 backbone to strengthen the extraction of slender structural patterns. This combination leads to a noticeable improvement in feature discrimination. We also adopt Focal Loss to reduce the impact of class imbalance during training. Putting all these pieces together results in more stable and robust detection performance. Overall, this design offers a tailored optimization strategy for underwater scenarios.

3.2. Evaluation metrics

In object detection, you need more than one metric to really tell how well a model is doing. For this study, we went with precision, recall, F1-score, and mAP. These four give us a solid basis for evaluating performance. Precision is calculated as follows.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

Here, TP stands for the number of positive detections the model got right. FP, on the other hand, is when negative samples are wrongly labeled as positive. That's a key distinction between the two different types of prediction errors. Recall is calculated as follows.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

FN here means the instances where positive samples are misclassified as negative, which captures a distinct form of prediction error.

Each category has its own precision and recall numbers. You can plot those to draw a precision–recall (P–R) curve. The area under that curve is called average precision (AP), and it tells you how accurate the detections are for that category. That's a pretty useful way to look at performance. Then, mean average precision (mAP) is simply the average of the AP values across all categories, which is given by the following formula.

$$\text{mAP} = \frac{\sum_1^N AP}{N} \tag{10}$$

Here, N stands for the total number of classes, which serves as a handy normalization term.

4. Experiment and Results

4.1. Experimental platforms and model's hyperparameters

During the training of the object detection model, its performance may vary depending on the experimental platform used. The hardware and software specifications employed in this study are listed in Table 1. Moreover, since hyperparameters can substantially influence model performance, we provide the full training configuration to ensure reproducibility. In our modified YOLOv8 [1–3] framework, the network was trained for 200 epochs with a batch size of 16. Prior to training, all images were uniformly resized to 640 × 640 pixels. The optimizer was configured using the "auto" setting, the random seed was fixed at 0, the close_mosaic value was set to 10, and the learning rate was initialized at 0.01. During training, the online data

Table 1
Computer parameters

Configuration	Parameter
Operating system	Ubuntu
Accelerated environment	Cuda 11.3
Language	Python 11.3
Framework	Pytorch 1.11.0
CPU	24 vCPU AMD EPYC 7642 48-Core Processor
GPU	RTX 3090 (24 GB)

augmentation technique we used was mosaic, which was disabled during the last 10 epochs of training.

4.2. Data augmentation results

Given the widespread presence of elongated structures in the dataset, we firmly believed that DSC would positively impact the results. However, our tests revealed that the performance of the deep learning model with added DSC showed almost no improvement compared to the model without DSC (Table 2).

Table 2
Comparison experiment without data augmentation

Methods	Precision (%)	Recall (%)	mAP50 (%)
YOLOv8	83.01	73.60	81.30
YOLOv8 + DSC	81.79	74.48	81.57

We attribute this result to the distortion and blurring phenomena in underwater images, which make the recognition of underwater targets more challenging. Consequently, DSC struggled to effectively capture the elongated structures in the data and efficiently learn the dataset's features, resulting in minimal performance improvement. Therefore, we applied the CLAHE algorithm to the dataset for data augmentation. Figure 8 shows the original images and the images after data augmentation.

As shown in Figure 8, after applying CLAHE data augmentation, the distortion and blurring phenomena present in the underwater images are effectively alleviated. The slender structures within the target objects become more clearly observable to the human eye, which suggests that further adding the DSC module will have a beneficial impact.

4.3. Effect of dynamic snake convolution

After we augmented the dataset, we plugged the DSC module into YOLOv8. Table 3 shows the final results.

Table 3 shows that incorporating the DSC module into the deep learning model and training it led to increases in precision, recall, F1-score, and mAP50, with performance gains exceeding those achieved without applying data augmentation. These results validate our hypothesis that high-quality, clear underwater images facilitate the deep learning model with the DSC module to capture elongated structures in the data more effectively and conduct detailed learning. Previous research in computer vision has shown the phenomenon of "Look Closer To See Better" [47], and here we observe a similar effect of "Look Clearer To Learn Better" [48] in computer vision as well.

Table 3
Comparison experiment after data augmentation

Methods	Precision (%)	Recall (%)	mAP50 (%)
YOLOv8	80.95	76.24	82.43
YOLOv8 + DSC	83.34	77.08	83.43

4.4. Comparison experiment of multiple different loss functions

The loss function quantifies the difference between the predicted output and the true value. A lower loss function indicates a better match between the predicted box and the true box. In our experiments, in addition to augmenting data and adding the DSC module to learn fine elongated features, we aim to improve the alignment between predicted and true boxes. Therefore, we experimented with various loss functions on our previous deep learning model, and the experimental results are presented in Table 4.

The deep learning model that incorporated the CIOU loss function achieved the highest precision value, while the upgraded version of CIOU, the EIOU loss function, consistently showed the lowest performance across all three metrics. We suspect this discrepancy is due to the additional parameters considered by EIOU not being suitable for our dataset. The deep learning model using the Focal loss function, named SDDL, achieved the best recall and mAP50 values by addressing class imbalance through penalty. To show how different loss functions change during training, we tracked the box_loss values throughout the training process, as depicted in Figure 9. It can be observed that, compared with the other loss functions, Focal Loss produces a lower loss value. Since

Figure 8
(a) Images without data augmentation and (b) images with data augmentation

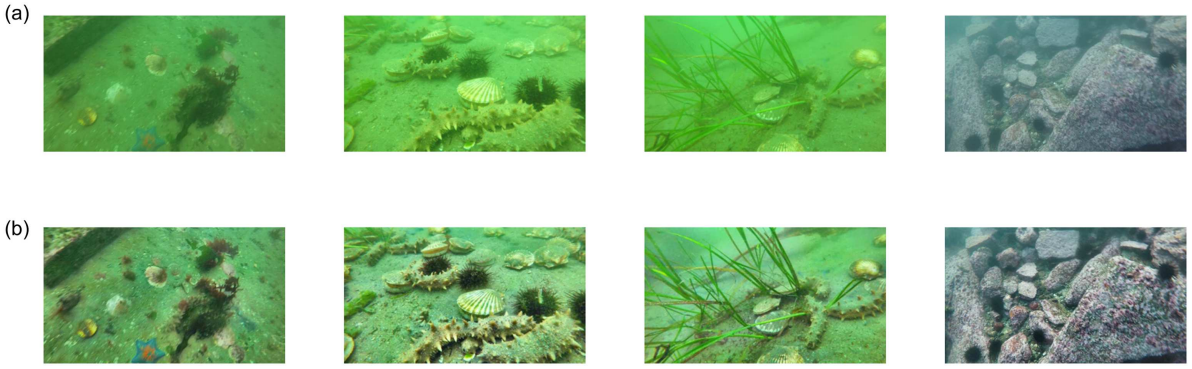
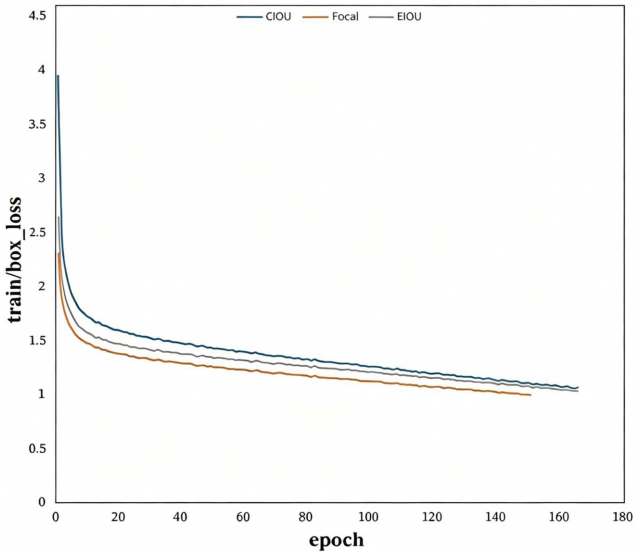


Table 4
Experimental results of various loss functions

Methods	Precision (%)	Recall (%)	mAP50 (%)
YOLOv8 + DSC + CIOU	83.34	77.08	83.43
YOLOv8 + DSC + EIOU	83.29	77.25	83.78
YOLOv8 + DSC + Focal	81.90	78.00	84.22

Figure 9
The changes in box_loss across different loss functions



the loss reflects the discrepancy between the predictions and the ground truth, this indicates that Focal Loss performs better and is more suitable for our dataset.

4.5. Ablation experiment

To evaluate the effectiveness of the proposed approach, CLAHE data augmentation, the DSC module, and the Focal loss function were incorporated into the baseline YOLOv8 model. The results, summarized in Table 5, indicate that our SDDL deep network achieved the highest scores in both Recall and mAP metrics. However, its precision value was lower compared to the highest value achieved by the YOLOv8 + CLAHE + DSC configuration. The table indicates that continuously adding CLAHE data augmentation, the DSC neural network module, and the Focal loss function to the baseline YOLOv8 model gradually increased the mAP value, thereby proving the effectiveness of our method. YOLOv8+DSC appears to perform better than YOLOv8+CLAHE+DSC on some metrics. However, under the more stringent localization-focused metric mAP50–95, our method achieves a 1.4% improvement. Moreover, compared with the baseline YOLOv8, our method improves all metrics except for a slight decrease in recall. For the ablation study results, we believe that the original images lack sufficient detail. After applying CLAHE, the image details become more pronounced, which allows the slender-feature extraction capability of DSC to be more effective. Finally, the use of Focal Loss alleviates the class imbalance in the dataset, resulting in the best overall performance.

4.6. Comparison with other detectors

To evaluate the effectiveness of our model, we conducted comparative experiments with several other detectors. The results are presented in Table 6. As shown, our model achieves the best performance on three metrics and the second-best on one metric. RetinaNet attains the highest Precision (92.5%); however, its performance on the remaining three metrics is considerably lower. We attribute this to its limited robustness when dealing with the significant noise commonly present in underwater environments.

RT-DETR-L, a transformer-based detector, was less accurate than our model and also required nine hours of training time, whereas ours finished in five hours. That difference really matters when resources are tight. And when we lined up our model against other YOLO variants, they all came out less accurate as well. So the evidence points pretty clearly to our method being superior.

4.7. The results of the SDDL algorithm applied to the seafood dataset

Underwater images often suffer from distortion and blurring caused by light attenuation and scattering, which can result in false positives and missed detections in seafood detection tasks. Additionally, unclear images can cause the targets to blend with the background, making them difficult to distinguish. As a result, the confidence scores produced by deep learning models tend to be lower. As shown in Figure 10, after our improvements, the greenish tint present in the original image has been alleviated, false detections have been reduced, model confidence has increased, and the number of missed detections has decreased. This facilitates underwater robots in using mechanical arms to grasp seafood with higher success rates and improves operational efficiency. Finally, in Figure 11, we present some of the results of seafood detection. It can be observed that most detection results have relatively high

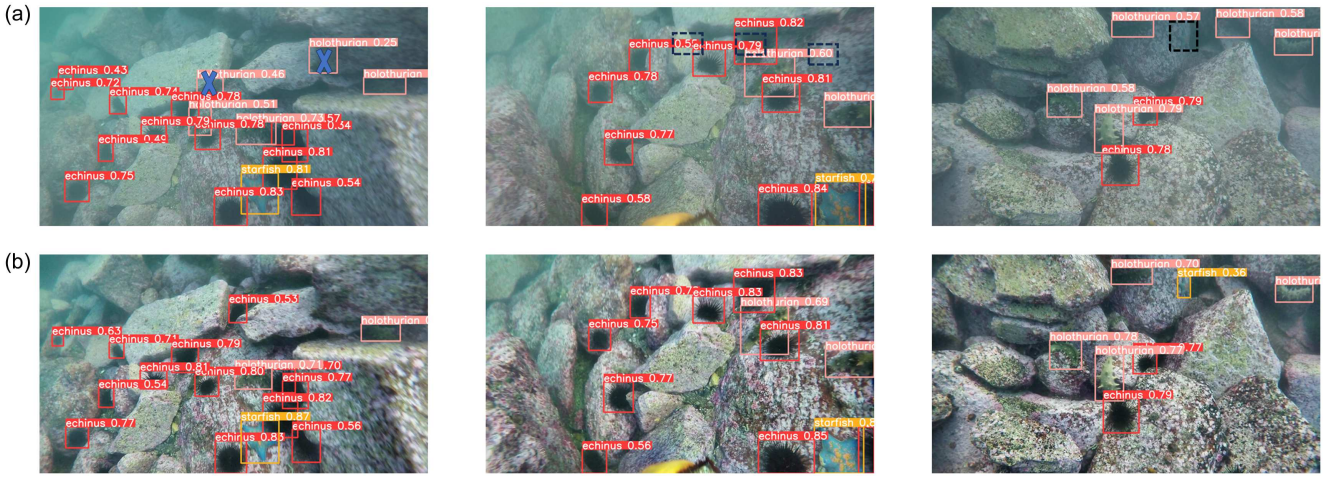
Table 5
Ablation experiment

Methods	Precision (%)	Recall (%)	mAP50 (%)	mAP50–95 (%)
YOLOv8	83.01	77.60	81.30	46.85
YOLOv8 + CLAHE	80.95	76.24	82.43	47.33
YOLOv8 + DSC	81.90	78.00	84.22	46.96
YOLOv8 + CLAHE + DSC	83.34	77.08	83.83	48.36
YOLOv8 + CLAHE + DSC + Focal	81.90	78.00	84.22	48.74

Table 6
Comparison experiment

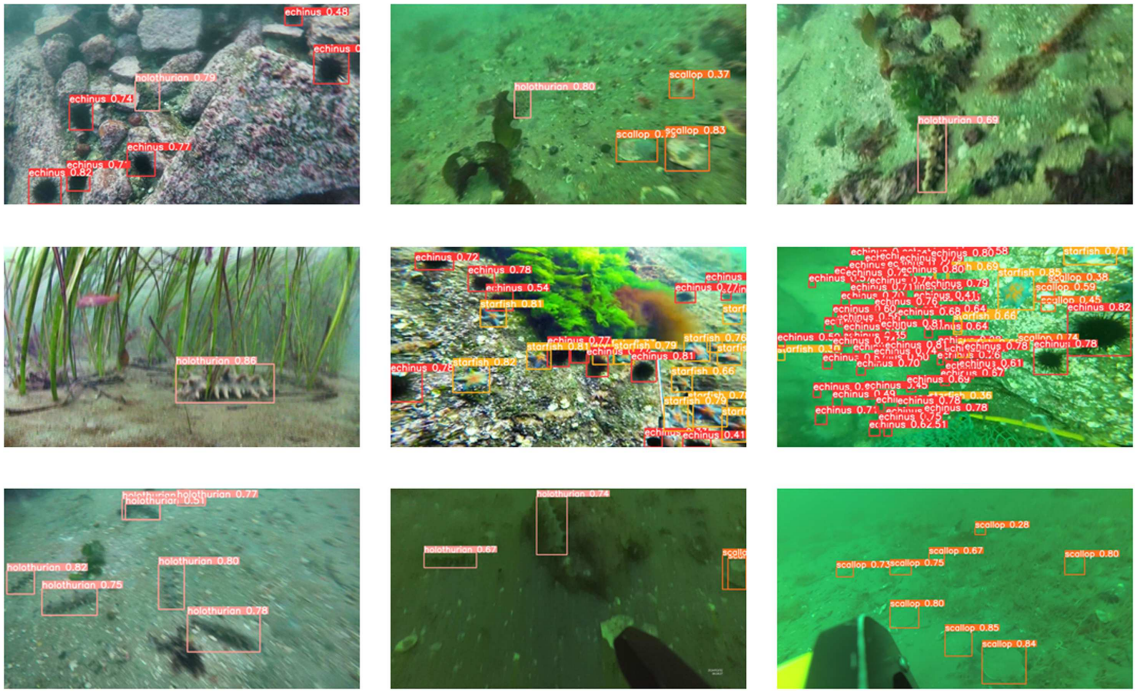
Methods	Precision (%)	Recall (%)	mAP50 (%)	mAP50–95 (%)
YOLOv5n	80.3	73.2	80.3	45.2
YOLOv6n	79.8	72	79.3	44.4
YOLOv7	79.3	73.9	80.4	42.8
YOLOv9t	81.4	74.2	81.7	47.2
YOLOv10n	79.1	74.4	80.6	46
YOLOv11n	79.5	75.3	81.3	46.1
RTDETR-L	77.6	70.5	77.2	42.4
Retinanet	92.5	51.7	72.3	39.1
Ours	81.9	78	84.2	48.7

Figure 10
(a) Detection results before improvement and (b) SDDL detection results after improvement



Note: The first column shows the comparison of false positives, the second column shows the comparison of confidence scores, and the third column shows the comparison of missed detections.

Figure 11
Seafood detection results display



confidence scores, generally above 0.5, indicating that the model is highly confident in its correct predictions. However, both missed detections and false detections still occur. In the first image of the second column, a scallop that is partially buried in the sediment and only partially visible is not detected, representing a missed detection. In the last image of the third column, the detection at the bottom has a low confidence score; however, the detected region is merely a shadow rather than an actual object, resulting in a false detection.

5. Conclusion and Limitations

In this study, we built a seafood detection model aimed at handling messy underwater environments. We did it by bringing in some data augmentation strategies. The model is specifically designed to pick up on the long, skinny shapes that a lot of seafood targets have. That gives it a real edge in capturing structural features. Here's a quick summary of what we found.

Underwater images are often distorted or blurry, which makes it hard to pick out the targets you care about. So we used a two-step approach: first, clean up the images and then run the model. That tackles a real problem caused by poor-quality input. The enhanced images then feed into our improved detection model. The whole process makes things visually clearer and helps the model extract features more reliably—basically a solid boost in data quality. In this work, we used CLAHE as part of data augmentation to strengthen contrast and make different seafood items stand out better. We also incorporated DSC to more effectively capture those long, skinny structural patterns. That combination makes the model particularly sensitive to structural features. On top of that, we applied Focal Loss to deal with class imbalance in multi-class seafood detection. This loss function tones down the influence of majority classes and pays more attention to the minority ones—a meaningful tweak to how training works. Experimental results show that our model delivers accurate seafood detection even under challenging conditions.

Our model hits 84.22% mAP and 48.74% mAP50–95, which is good enough for real-world fisheries and aquaculture applications [1, 2]. These numbers show just how much better image quality and a network designed for underwater conditions can boost recognition performance. It's a solid step forward for application-oriented detection. The model can be deployed on underwater robots to identify seafood targets in marine environments. That kind of integration supports real-world automated operations—a meaningful move toward practical deployment. Going forward, we plan to explore lightweight model architectures and build a cloud-based database using the detection data we collect. These directions should help improve both the efficiency and scalability of the overall system, moving us toward more optimized and distributed solutions.

Our findings are pretty consistent with what earlier studies have observed. Cleaning up underwater images—making them clearer—does help neural networks detect things better. So there's a real connection between image quality and how well a model performs. DSC, for its part, helps pull out more representative structural features. And Focal Loss boosts accuracy by dialing down the impact of class imbalance. Each component in the model contributes in its own way. Taken together, the results we've presented here should serve as a useful reference for future work on underwater image processing and detection tasks.

That said, our study does have a few limitations. The dataset only covers four types of seafood, so future work could expand it to include more species. That's a real constraint on diversity. Also, the range of underwater conditions we tested is still fairly narrow—we haven't run the model on a broader set of datasets yet. So there's a clear gap in validation coverage that needs to be filled.

In future work, we'll look at how the model performs across a wider range of conditions and datasets, which will give us a much more thorough assessment.

Acknowledgment

I sincerely thank the editor and anonymous reviewers for their valuable suggestions, which helped improve the quality of this manuscript. I am grateful to Professor Zhao Wen tao for providing research funding; even though we did not work closely together, your guidance has been truly inspiring. Many thanks to Dr. Ma Fuyu for providing equipment and technical support and for accompanying me throughout my academic journey. Special appreciation also goes to my pet hamster, Little Lee-Yan—your cute, round presence brought endless joy, even if your occasional nibbles sometimes disrupted my work. I am deeply sorry for my negligence that allowed you to escape from the cage and fall into the toilet where you drowned. Whenever I think of this incident, I am overwhelmed with grief.

Funding Support

This work is supported by the National Natural Science Foundation of China (U23A20645).

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The URPC dataset that support the findings of this study is available in Github at https://github.com/mousecpn/DG-YOLO.

Author Contribution Statement

Hanyu Jiang: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Project administration, Funding acquisition. **Zhichao Zheng:** Conceptualization, Methodology, Software, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Supervision.

References

[1] Chen, G., Mao, Z., Tu, Q., & Shen, J. (2024). A cooperative training framework for underwater object detection on a clearer view. *IEEE Transactions on Geoscience and Remote Sensing*, *62*, 4208817. https://doi.org/10.1109/TGRS.2024.3440386

[2] Wang, B., Wang, Z., Guo, W., & Wang, Y. (2024). A dual-branch joint learning network for underwater object detection.

Knowledge-Based Systems, *293*, 111672. https://doi.org/10.1016/j.knosys.2024.111672

[3] Dai, L., Liu, H., Song, P., & Liu, M. (2024). A gated cross-domain collaborative network for underwater object detection. *Pattern Recognition*, *149*, 110222. https://doi.org/10.1016/j.patcog.2023.110222

[4] Strachan, N. J. C., Nesvadba, P., & Allen, A. R. (1990). Fish species recognition by shape analysis of images. *Pattern Recognition*, *23*(5), 539–544. https://doi.org/10.1016/0031-3203(90)90074-U

[5] Spampinato, C., Giordano, D., di Salvo, R., Chen-Burger, Y.-H. J., Fisher, R. B., & Nadarajan, G. (2010). Automatic fish classification for underwater species behavior understanding. In *Proceedings of the First ACM International Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams*, 45–50. https://doi.org/10.1145/1877868.1877881

[6] Toh, Y. H., Ng, T. M., & Liew, B. K. (2009). Automated fish counting using image processing. In *2009 International Conference on Computational Intelligence and Software Engineering*, 1–5. https://doi.org/10.1109/CISE.2009.5365104

[7] Fouad, M. M. M., Zawbaa, H. M., El-Bendary, N., & Hassanien, A. E. (2013). Automatic Nile Tilapia fish classification approach using machine learning techniques. In *13th International Conference on Hybrid Intelligent Systems*, 173–178. https://doi.org/10.1109/HIS.2013.6920477

[8] Yassir, A., Andaloussi, S. J., Ouchetto, O., Mamza, K., & Serghini, M. (2023). Acoustic fish species identification using deep learning and machine learning algorithms: A systematic review. *Fisheries Research*, *266*, 106790. https://doi.org/10.1016/j.fishres.2023.106790

[9] Paygude, P., Shinde, N., Dhumane, A., Navale, G. S., Chavan, P., Kathole, A., & Bidve, V. (2025). Species identification for Indian seafood markets: A machine learning approach with a fish dataset. *Data in Brief*, *58*, 111209. https://doi.org/10.1016/j.dib.2024.111209

[10] Deka, J., Laskar, S., & Baklial, B. (2023). Automated freshwater fish species classification using deep CNN. *Journal of The Institution of Engineers (India): Series B*, *104*(3), 603–621. https://doi.org/10.1007/s40031-023-00883-2

[11] Ovalle, J. C., Vilas, C., & Antelo, L. T. (2022). On the use of deep learning for fish species recognition and quantification on board fishing vessels. *Marine Policy*, *139*, 105015. https://doi.org/10.1016/j.marpol.2022.105015

[12] Li, J., Xu, W., Deng, L., Xiao, Y., Han, Z., & Zheng, H. (2023). Deep learning for visual recognition and detection of aquatic animals: A review. *Reviews in Aquaculture*, *15*(2), 409–433. https://doi.org/10.1111/raq.12726

[13] Jiang, H., Zhao, J., Ma, F., Yang, Y., & Yi, R. (2025). Mobile-YOLO: A lightweight object detection algorithm for four categories of aquatic organisms. *Fishes*, *10*(7), 348. https://doi.org/10.3390/fishes10070348

[14] Jiang, H., Zhong, J., Ma, F., Wang, C., & Yi, R. (2025). Utilizing an enhanced YOLOv8 model for fishery detection. *Fishes*, *10*(2), 81. https://doi.org/10.3390/fishes10020081

[15] Jiang, H., Zhong, J., & Wang, C. (2025). Detection of cotton pests using an enhanced deep learning model. *Journal of Asia-Pacific Entomology*, *28*(3), 102450. https://doi.org/10.1016/j.aspen.2025.102450

[16] Zhang, S., Gong, M., Xie, Y., Qin, A. K., Li, H., Gao, Y., & Ong, Y.-S. (2022). Influence-aware attention networks for anomaly detection in surveillance videos. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(8), 5427–5437. https://doi.org/10.1109/TCSVT.2022.3148392

[17] Su, J., Su, Y., Zhang, Y., Yang, W., Huang, H., & Wu, Q. (2022). EpNet: Power lines foreign object detection with Edge Proposal Network and data composition. *Knowledge-Based Systems*, *249*, 108857. https://doi.org/10.1016/j.knosys.2022.108857

[18] Zhou, Y., Wang, F., Zhao, J., Yao, R., Chen, S., & Ma, H. (2022). Spatial-temporal based multihead self-attention for remote sensing image change detection. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(10), 6615–6626. https://doi.org/10.1109/TCSVT.2022.3176055

[19] Liao, G., Gao, W., Li, G., Wang, J., & Kwong, S. (2022). Cross-collaborative fusion-encoder network for robust RGB-thermal salient object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(11), 7646–7661. https://doi.org/10.1109/TCSVT.2022.3184840

[20] Chen, L., Huang, Y., Dong, J., Xu, Q., Kwong, S., Lu, H., . . . , & Li, C. (2026). Underwater optical object detection in the era of artificial intelligence: Current, challenge, and future. *ACM Computing Surveys*, *58*(3), 62. https://doi.org/10.1145/3759243

[21] Chen, L., Zhou, F., Wang, S., Dong, J., Li, N., Ma, H., . . . , & Zhou, H. (2022). SWIPENET: Object detection in noisy underwater scenes. *Pattern Recognition*, *132*, 108926. https://doi.org/10.1016/j.patcog.2022.108926

[22] Song, P., Li, P., Dai, L., Wang, T., & Chen, Z. (2023). Boosting R-CNN: Reweighting R-CNN samples by RPN's error for underwater object detection. *Neurocomputing*, *530*, 150–164. https://doi.org/10.1016/j.neucom.2023.01.088

[23] Dai, L., Liu, H., Song, P., Tang, H., Ding, R., & Li, S. (2024). Edge-guided representation learning for underwater object detection. *CAAI Transactions on Intelligence Technology*, *9*(5), 1078–1091. https://doi.org/10.1049/cit2.12325

[24] Shen, X., Wang, H., Cui, T., Guo, Z., & Fu, X. (2024). Multiple information perception-based attention in YOLO for underwater object detection. *The Visual Computer*, *40*(3), 1415–1438. https://doi.org/10.1007/s00371-023-02858-2

[25] Shen, X., Wang, H., Li, Y., Gao, T., & Fu, X. (2024). Criss-cross global interaction-based selective attention in YOLO for underwater object detection. *Multimedia Tools and Applications*, *83*(7), 20003–20032. https://doi.org/10.1007/s11042-023-16311-y

[26] Liang, X., & Song, P. (2022). Excavating RoI attention for underwater object detection. In *2022 IEEE International Conference on Image Processing*, 2651–2655. https://doi.org/10.1109/ICIP46576.2022.9897515

[27] Shen, X., Sun, X., Wang, H., & Fu, X. (2023). Multi-dimensional, multi-functional and multi-level attention in YOLO for underwater object detection. *Neural Computing and Applications*, *35*(27), 19935–19960. https://doi.org/10.1007/s00521-023-08781-w

[28] Mumuni, A., & Mumuni, F. (2022). Data augmentation: A comprehensive survey of modern approaches. *Array*, *16*, 100258. https://doi.org/10.1016/j.array.2022.100258

[29] Liu, C., Wang, Z., Wang, S., Tang, T., Tao, Y., Yang, C., . . . , & Fan, X. (2022). A new dataset, Poisson GAN and AquaNet for underwater object grabbing. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(5), 2831–2844. https://doi.org/10.1109/TCSVT.2021.3100059

[30] Liu, R., Jiang, Z., Yang, S., & Fan, X. (2022). Twin adversarial contrastive learning for underwater image enhancement

and beyond. *IEEE Transactions on Image Processing*, *31*, 4922–4936. https://doi.org/10.1109/TIP.2022.3190209

[31] Fu, C., Fan, X., Xiao, J., Yuan, W., Liu, R., & Luo, Z. (2023). Learning heavily-degraded prior for underwater object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, *33*(11), 6887–6896. https://doi.org/10.1109/TCSVT.2023.3271644

[32] Xu, S., Zhang, M., Song, W., Mei, H., He, Q., & Liotta, A. (2023). A systematic review and analysis of deep learning-based underwater object detection. *Neurocomputing*, *527*, 204–232. https://doi.org/10.1016/j.neucom.2023.01.056

[33] Das, A., Pathan, F., Jim, J. R., Kabir, M. M., & Mridha, M. F. (2025). Deep learning-based classification, detection, and segmentation of tomato leaf diseases: A state-of-the-art review. *Artificial Intelligence in Agriculture*, *15*(2), 192–220. https://doi.org/10.1016/j.aiia.2025.02.006

[34] Hu, J., Bao, C., Ozay, M., Fan, C., Gao, Q., Liu, H., & Lam, T. L. (2023). Deep depth completion from extremely sparse data: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(7), 8244–8264. https://doi.org/10.1109/TPAMI.2022.3229090

[35] Liu, Y., Li, H., Hu, C., Luo, S., Luo, Y., & Chen, C. W. (2025). Learning to aggregate multi-scale context for instance segmentation in remote sensing images. *IEEE Transactions on Neural Networks and Learning Systems*, *36*(1), 595–609. https://doi.org/10.1109/TNNLS.2023.3336563

[36] Qi, Y., He, Y., Qi, X., Zhang, Y., & Yang, G. (2023). Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation. In *2023 IEEE/CVF International Conference on Computer Vision*, 6047–6056. https://doi.org/10.1109/ICCV51070.2023.00558

[37] Wang, X., & Song, J. (2021). ICIoU: Improved loss based on complete intersection over union for bounding box regression. *IEEE Access*, *9*, 105686–105695. https://doi.org/10.1109/ACCESS.2021.3100414

[38] Wang, C.-Y., Yeh, I.-H., & Mark Liao, H.-Y. (2025). YOLOv9: Learning what you want to learn using programmable gradient information. In *Computer Vision – ECCV 2024: 18th European Conference*, 1–21. https://doi.org/10.1007/978-3-031-72751-1_1

[39] Yang, Z., Wang, X., & Li, J. (2021). *EIoU*: An improved vehicle detection algorithm based on VehicleNet neural network. *Journal of Physics: Conference Series*, *1924*(1), 012001. https://doi.org/10.1088/1742-6596/1924/1/012001

[40] Ali, M. L., & Zhang, Z. (2024). The YOLO framework: A comprehensive review of evolution, applications, and benchmarks in object detection. *Computers*, *13*(12), 336. https://doi.org/10.3390/computers13120336

[41] Kamdan, Firdaus, A. R., Nasrulloh, D., Tresna Ati, M. A., Kharisma, I. L., & Sujjada, A. (2024). Performance analysis of the YOLOv5 model in planthopper pest detection: From Nano to Medium. In *2024 10th International Conference on Computing, Engineering and Design*, 1–7. https://doi.org/10.1109/ICCED64257.2024.10983691

[42] Guo, C., Lv, X., Zhang, Y., & Zhang, M. (2021). Improved YOLOv4-tiny network for real-time electronic component detection. *Scientific Reports*, *11*(1), 22744. https://doi.org/10.1038/s41598-021-02225-y

[43] Li, H., Gu, Z., He, D., Wang, X., Huang, J., Mo, Y., . . . , & Wu, F. (2024). A lightweight improved YOLOv5s model and its deployment for detecting pitaya fruits in daytime and nighttime light-supplement environments. *Computers and Electronics in Agriculture*, *220*, 108914. https://doi.org/10.1016/j.compag.2024.108914

[44] Hussain, A., Li, H.-C., Ali, D., Ali, M., Abbas, F., & Hussain, M. (2023). An optimized deep supervised hashing model for fast image retrieval. *Image and Vision Computing*, *133*, 104668. https://doi.org/10.1016/j.imavis.2023.104668

[45] Hussain, A., Li, H.-C., Ali, M., Wali, S., Hussain, M., & Rehman, A. (2022). An efficient supervised deep hashing method for image retrieval. *Entropy*, *24*(10), 1425. https://doi.org/10.3390/e24101425

[46] Hussain, A., li, H.-C., Hussain, M., Ali, M., Abbas, S., Ali, D., & Rehman, A. (2024). A gradual approach to knowledge distillation in deep supervised hashing for large-scale image retrieval. *Computers and Electrical Engineering*, *120*, 109799. https://doi.org/10.1016/j.compeleceng.2024.109799

[47] Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., . . . , & Hu, S.-M. (2022). Attention mechanisms in computer vision: A survey. *Computational Visual Media*, *8*(3), 331–368. https://doi.org/10.1007/s41095-022-0271-y

[48] Fayaz, S., Parah, S. A., Qureshi, G. J., Lloret, J., del Ser, J., & Muhammad, K. (2024). Intelligent underwater object detection and image restoration for autonomous underwater vehicles. *IEEE Transactions on Vehicular Technology*, *73*(2), 1726–1735. https://doi.org/10.1109/TVT.2023.3318629

How to Cite: Jiang, H., & Zheng, Z. (2026). Underwater Seafood Detection Using Deep Learning and Data Augmentation. *Journal of Data Science and Intelligent Systems*. https://doi.org/10.47852/bonviewJDSIS62027779