IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 19 April 2024, accepted 16 May 2024, date of publication 24 May 2024, date of current version 28 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3405452

RESEARCH ARTICLE

Improved YOLOv5 Based Deep Learning System for Jellyfish Detection

THI-NGOT PHAM[1,2,*], VIET-HOAN NGUYEN[3,4,*], KI-RYONG KWON[3], (Member, IEEE), JAE-HWAN KIM[5], AND JUN-HO HUH[2,5], (Member, IEEE)

[1]Department of Data Informatics, National Korea Maritime and Ocean University, Busan 49112, Republic of Korea
[2]Interdisciplinary Major of Ocean Renewable Energy Engineering, National Korea Maritime and Ocean University, Busan 49112, Republic of Korea
[3]Department of Artificial Intelligence Convergence, Pukyong National University, Busan 48513, Republic of Korea
[4]Intown Company Ltd., Busan 46207, Republic of Korea
[5]Department of Data Science, National Korea Maritime and Ocean University, Busan 49112, Republic of Korea

Corresponding authors: Jun-Ho Huh (72networks@kmou.ac.kr) and Jae-Hwan Kim (jhkim@kmou.ac.kr)

This work was supported in part by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program, supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP), under Grant IITP-2024-2020-0-01797; and in part by the 2023 BK21 FOUR Graduate School Innovation Support funded by Korea Maritime and Ocean University (KMOU Graduate School Innovation—Fellowship Program).

*Thi-Ngot Pham and Viet-Hoan Nguyen contributed equally to this work.

ABSTRACT Massive jellyfish outbreaks have put human lives and marine ecosystems in great danger. As a result, the jellyfish detection methods have drawn a lot of attention, following two directions optical and sonar imaging. This work focuses on using optical imagery and CNN-based deep-learning object detection models to detect jellyfish. While labeled data of jellyfish play an important part in training deep learning models, there are a few open and available labeled datasets. Hence, we create our dataset to train these models using our model-assisted labeling method with over 11 thousand images of underwater jellyfish and corresponding annotation files in PASCAL VOC format. Our model-assisted labeling method saves the work of classical manual labeling by 70 percent, which is developed into application with YOLOv5. However, the YOLOv5 baseline suffers from the trade-off between real-time performance and low accuracy. Hence, an improved YOLOv5-nano is introduced based on adding GAM and replacing conventional Conv with CoordCov modules into the backbone of the conventional structure. The experiment results show that our improved model increases the accuracy of the conventional one by 1.3% and outperforms others including RetinaNet, SSD, Faster R-CNN, YOLOv6, and YOLOv8 at 89.1% mAP@0.5. On generalization performance, we verify the effectiveness of our work by conducting a test set of 15 different types of jellyfish with various shapes, colors, resolutions, and backgrounds. To conclude, our work establishes a comprehensive system from labeling the data, improving object detectors, and developing a feasible real-time jellyfish detector.

INDEX TERMS Jellyfish detection, deep learning, YOLOv5, coordinate attention, GAM, CoordCov.

I. INTRODUCTION

Jellyfish, which make up a significant portion of marine plankton biomass, are mostly carnivorous and eat a variety of different foods. As a result, the outbreaks of jellyfish frequently result in a dramatic reduction in zooplankton density, which causes fish to starve to death and a decline in fish stock density [1]. Many maritime locations, for example, Masan Bay in Korea, have had a jellyfish outbreak in recent years as a result of a significant number of jellyfish breeding, which has severely damaged the local ecology and economy [2]. These jellyfish outbreaks have a significant impact on both the sustainability of the maritime economy and the ecological security of the coastal region. Therefore, it is essential to monitor the jellyfish species, location, and jellyfish bloom distribution.

The associate editor coordinating the review of this manuscript and approving it for publication was Junchi Yan.

Optical imaging and sonar imaging are the most advanced underwater imaging techniques. By sending and receiving sonar waves, underwater acoustic imaging enables the identification of jellyfish but has the drawback of having limited picture quality. Because optical imaging has a better resolution than underwater acoustic imaging, it has quickly advanced in detecting underwater objects. According to literature reviews, the current trend of detection method is a convolutional neural network (CNN)-based deep learning objection detection, which has progressively become a cutting-edge method to research marine species (i.e., fish, shrimp, and jellyfish).

Hence, in this work, we focus on the jellyfish detection approach using optical underwater images and CNN-based deep-learning object detectors. In [3], [4], [5], [6], and [7], the development of object detection has passed by two milestones: before 2014 as the traditional models' period, and after 2014 as the deep learning models era. Deep learning object detection can be categorized into two groups: region recommendations as "two-stage detector" (i.e., Recurrent Convolutional Neural Network (R-CNN) (2014) [8], Fast R-CNN (2015) [9], and Faster R-CNN (2015) [10]) and regression-based algorithms as "one-stage detector" (i.e., You Only Look One (YOLO) (2016) [11], Single Shot Multi-Box Detector (SSD) (2016) [12], Retina-Net (2018) [13]). YOLO-family has a long history of development, and the most popular versions up till now include YOLOv4-Darknet (2020) [14], YOLOv5 by Ultralytics [15], YOLOv6 by Meituan (2023) [16], YOLOv8 by Ultralytics [17].

In the case of jellyfish detectors, these two approaches have drawn the attention of many researchers. Since 2016, CNN-deep learning-based object detectors have been applied for jellyfish detection. During the period from 2016 to 2021, Faster R-CNN with different kinds of backbone (i.e., Inception, ResNet, AlexNet, GoogleNet) showed their dominance compared to YOLO-models (i.e., YOLO, YOLOv2). However, after its release in 2020, YOLOv4 and advanced YOLO versions have gradually replaced the dominance of Faster R-CNN with its real-time performance and high accuracy. There were previous studies related to improved YOLOv3 [18], YOLOv4, and its lightweight version YOLOv4-Tiny for jellyfish detection. Apart from detectors, labeled training data play an important part in the success of deep learning-based object detectors. However, there are a few open datasets with annotations [19] because the manual annotation process takes a lot of time and labor to label thousands of images. As in [15], to get a better classification result, each class consists of over 2 thousand images in case of a single label per image (a total of 2 thousand labels per class).

As a result, instead of following classical manual annotation methods, we introduce a model-assisted annotation method with auto-labeling, which we published in [20], to create our jellyfish dataset to save time and labor of annotation task. Therefore, the model deployed into the auto-labeling module requires fast detection time with an accurate detection rate without missing, over-detection, and better localization of bounding boxes around jellyfish. In this work, we focus on the latter YOLO models and their lightweight version, particularly YOLO version 5 nano (YOLOv5n). The main idea of applying the nano version is to keep fast inference speed, but there is a drawback of low accuracy that needs to be solved. To solve the accuracy issue, adding an attention mechanism module into the baseline network or replacing the conventional module with advanced ones are the most preferable solutions. The contribution of our work is presented as follows:

- We introduce a new dataset with a total of 14376 images and corresponding labels by applying our model-assisted annotation application.
- We propose an improved YOLOv5n by adding a Global Attention Mechanism (GAM) into the baseline network to enhance its feature extraction capability boost the small object and overlapping object detection rate and replace the Convolutional layer with Coordinate Convolutional layer (CoordConv) to improve its IoU localization accuracy.

II. RELATED WORKS

With the development of science and technology, researchers use various methods to detect underwater creatures, such as optical imaging, sonar imaging, remote sensing sensors, etc. Table 1 describes our summary of significant breakthroughs in jellyfish detection, classification, and segmentation using CNN-based deep learning models.

TABLE 1. Related works.

References	Datasets	Methods
Hangeun Kim et al (2016)	Water surface optical image	Proposed CNN
Jungmo Koo et al (2017)	Water surface Optical image	Proposed CNN, LeNet-5, AlexNet, GoogleNet
French et al (2018)	Underwater sonar image	VGG-16
Martin-Abadal et al (2020)	Underwater optical image	Faster R-CNN (Inception-ResNetV2, Inception V2, ResNet101)
Ying Han et al (2021)	Underwater optical image 10 types of jellyfish & 1 type of fish	Faster R-CNN (AlexNet and GoogleNet)
Chang Quiyue et al (2021)	Underwater optical image 7 types of jellyfish 2141 images	Improved YOLOv3
Meijing Gao et al (2021)	Underwater optical image 7 types of jellyfish and 1 type of fish 11926 images	Improved YOLOv3
Meijing Gao et al (2023)	Underwater optical image 7 types of jellyfish and 1 type of fish	Improve YOLOv4-Tiny +CBAM
Wengming Zhang et al (2023)	Optical image on water surface	Improved YOLOv4

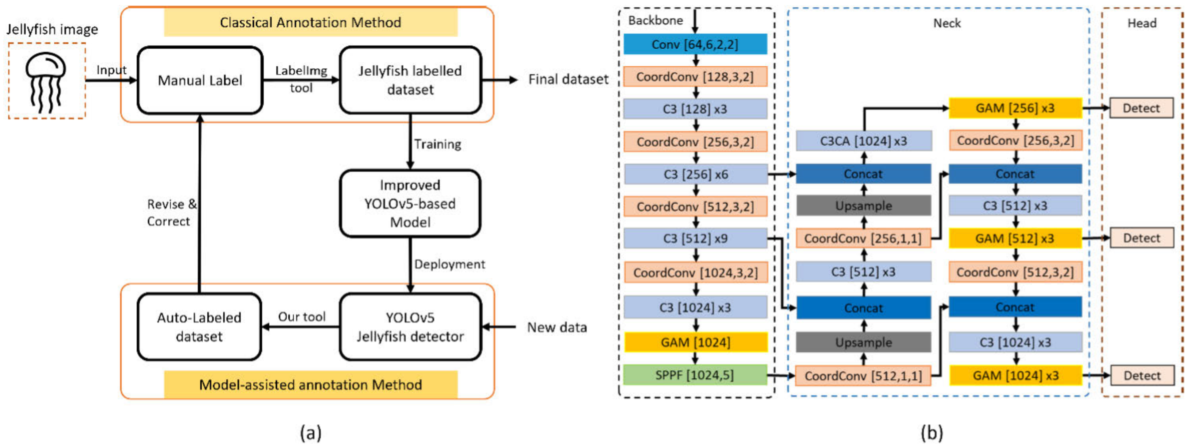

FIGURE 1. **The deep learning system for jellyfish detection: (a) System design, (b) Network structure of improved YOLOv5-based model.**

In 2016, a jellyfish removal system was proposed by Kim et al. [21], which utilized unmanned aerial vehicles (UAV) to segment jellyfish on the water surface using their proposed CNN-based deep learning model. In 2017, a jellyfish distribution recognition system was developed by Koo et al. [22] employing an unmanned aerial vehicle (UAV) and an unmanned surface vehicle (USV). A CNN deep learning-based model based on LeNet5 [23] and AlexNet [24] was proposed for jellyfish segmentation on the surface of seawater, which outperformed the baseline and GoogleNet [25] on experiment results. However, these studies have constraints of jellyfish segmentation on the water surface, which is unable to detect underwater jellyfish. In 2018, French et al. [26] employed a CNN-based VGG-16 [27] classifier to detect jellyfish versus 5 other classes (i.e., background, sediment, artifacts, fish, and seaweed) using underwater sonar imaging.

Martin-Abadal et al. [28] introduced Jellytoring a jellyfish monitoring based on Faster R-CNN with different back-bone of Inception-ResNetV2 [29], Inception V2 [30], ResNet101 [31], which can detect 3 types of jellyfish, and record the existence of jellyfish for a long time. It is noted that their dataset was open and available with labeled annotations [19]. Han et al. [32] introduced jellyfish classification and detection based on Faster R-CNN with two cases of different backbones, namely AlexNet and GoogleNet. From their experiment results, the Faster R-CNN algorithm based on GoogLeNet outperformed AlexNet with an accuracy of 74.96%.

In 2021, Qiuyue et al. [33] proposed an improved YOLOv3 which classified seven jellyfish species. Gao et al. [34], [35] proposed an improved YOLOv3 and YOLOv4-Tiny for jellyfish classification and detection respectively. As can be seen from previous studies, from 2021, there is a shifting trend from a two-stage detector (i.e., Faster RCNN with GoogleNet, AlexNet, ResNet101 models) to a one-stage detector (i.e., YOLOv3, YOLOv4 models). The previous studies provide some insights into our work but there is still room for improvement. For example, the previous versions of YOLO (i.e., YOLOv3 and YOLOv4) are quite out of date with the recent release of YOLOv5, YOLOv6, and YOLOv8.

The parameter of network and computation load FLOPs of YOLOv5 are the smallest compared to other YOLO models such as YOLOv6 and YOLOv8. Therefore, in this work, we propose an improved YOLOv5 model for jellyfish detection and creating a labeled jellyfish dataset. Our model-assisted annotation application is first developed based on customized YOLOv5 version 6.0 source code. In addition, adding attention mechanism modules into the YOLO conventional baseline network is a preferable approach.

However, apart from adding the Convolutional Block Attention Module (CBAM) in [36], there are some other noticeable algorithms and there is no evaluation of them for jellyfish detection task. Hence, to verify the effectiveness of adding an attention module, we deploy various modules into YOLOv5n baseline, including Coordinate Attention (CA) [37], CBAM, Triplet Attention Module (TAM) [38], Efficient Channel Attention (ECA) [39], and Global Attention Mechanism (GAM) [40] and conduct ablation studies. While the recent studies focused on multiple categories of jellyfish classification and detection, i.e., 10 types of jellyfish and 1 type of fish in [32], [34], and [35], our work aims to detect multiple jellyfish only without classification. In our training/evaluation set, there is only one class, "jellyfish". Our testing set consists of 15 different jellyfish species to verify our improved YOLOv5n model generalization performance.

III. METHODOLOGY

A. MODEL-ASSISTED ANNOTATION METHOD

A jellyfish dataset with a target number of images of about 10 thousand images is created in our work. Following the classical annotation methods, open-source LabelImg [41] is used to draw bounding boxes around jellyfish and set labels manually. However, labeling a thousand images task requires a lot of time and labor, to solve these issues, we introduce a model-assisted annotation method as shown in Fig. 1(a).

Firstly, the unlabeled images are split into 10 parts following the 80/20 percentage of training and validation set. The first part with 10 percent of training set (about 1400 images) will be labeled manually using labelImg tool to create 1^{st} trial small-scale jellyfish dataset. There are five steps of annotating with labelImg as can be seen in Fig. 2. After opening the direction of image folder in the first step, all the images inside folder are loaded into labelImg tool as shown in file list in the bottom right corner of GUI. To fasten the manual annotation, it is preferable to label images with a single big jellyfish first and utilize a model-assisted auto-label application to process images of multiple or small jellyfish in the next phase. Then, the most important step is step 2 "S2. Select annotation format". The labelImg tool supports three types of annotation formats, including YOLO, PascalVOC, and CreateML format. It is recommended to save annotation files as XML files in PascalVOC format to easily convert to other formats, including YOLO as txt files and COCO as json files. In step 3, it is better to draw the bounding boxes around the jellyfish as fit as possible. In "Edit labels" step, there is one label as "jellyfish" in our work, thus it is an easily set label. It is noted that there is a default label set consisting of all the names of labels, which are stored in label txt file in labelImg tool folder. Hence, in case of labeling a dataset with multiple classes, it is better to edit this file with the names of all classes in advance to fasten the set label process. In the last step of manual label "Saving", the XML files will be saved in the same folder as image directory.

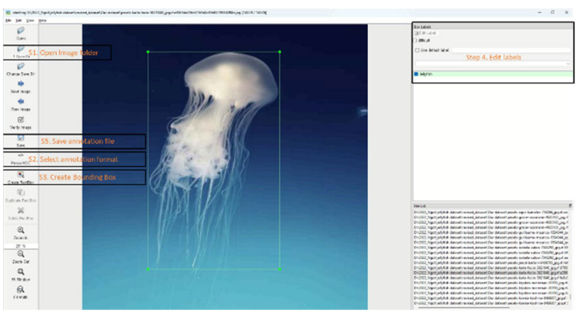

FIGURE 2. Manually annotate the image with LabelImg.

Secondly, the XML annotation files will be converted to YOLO format as txt files to train with pre-trained YOLOv5n weight "yolov5n.pt" using transfer learning. A single GPU RTX3050 is utilized on our deployment system, thus a lightweight model such as YOLO version 5 nano (YOLOv5n) is used to process this annotation task. After that, the trained weight will be used as the input of our model-assisted annotation application to label the remaining images. The remaining nine parts of unlabeled images will be fetched into YOLO jellyfish detector. Then, the bounding boxes and labels detected by YOLOv5 jellyfish detector are written automatically into ".xml" files using auto-labeling module. It is noted that the confidence threshold in YOLO jellyfish detector is set from 0.6 to 0.7. According to different inputs, in the GUI application, the detection results will be displayed so we can adjust the confidence threshold for better results. The motivation for setting high confidence is to reduce false detection. Only images consisting of detection results with high confidence and the corresponding XML files will be saved in the output directory.

However, the most common type of error using model-assisted labeling method is the mismatch of bounding boxes fitting to an object and the miss detection due to the overlapping of multiple objects or small objects. Hence, subsequently, these annotation files will be revised and corrected manually using labelImg tool. Fig. 3(a) and (b) present examples of before and after the process of revision and correcting in case of two common errors respectively. As can be seen in Fig. 3, the auto-label bounding boxes are re-drawn to fit the jellyfish in case of mismatch, and new bounding boxes are drawn in case of miss detection. Besides, labelImg supports duplication of existing bounding boxes instead of drawing a new one. In some cases of similar sizes of jellyfish, it is better to duplicate the bounding boxes.

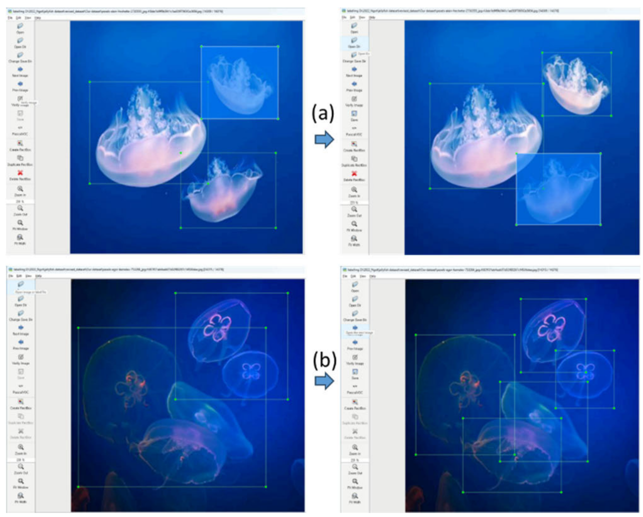

FIGURE 3. Example of revising and correcting manually in case of (a) bounding boxes mismatch fitting and (b) miss detection due to overlapping.

Finally, after revising and correcting the phase, the images and annotations files will be added to existing dataset to re-train again to update the trained weights following active learning. To fasten the re-training process with new images, it is recommended to load input data of training/validation set for YOLO model from reading existing files in folder instead of reading lines in txt files. For example, the training and validation set is configured in "data.yaml" file with the input "train: basepath/images/train/" instead of "train: basepath/train.txt". After each active learning with re-train, the updated weights will be deployed to YOLO detector to auto-label the remaining images. It is noted that after each round, the confidence threshold of YOLO detector will be reduced from 0.7 to lower confidence by 0.05 or depending on input data as our model-assisted labeling app supports adjusting confidence threshold and displaying real-time detection results on GUI. Due to the remaining images with many small or overlapping jellyfish, reducing the confidence threshold, increases the detection rates. Also, due to experience with some drawbacks of YOLO baseline, we introduce an improved YOLO version to solve these issues. Then, the process of revising and correcting will be processed several times until completion of our dataset.

The UML of our model-assisted annotation application is presented in Fig. 4, including three main steps: optional selections (model, keyword, data file), trained model detection (YOLOv5 detector), and auto-labeling. It is noted that the YOLOv5 jellyfish detector source code is customized based on Ultralytics [15], while auto-labeling module and other parts are developed by our previous work [20]. As can be seen on the optional selections, there are several functions to select the inserting YOLOv5 jellyfish detector, including selection model, data collection, and the selection of a single file or multiple files. After finishing selecting the input data and model, the YOLOv5 jellyfish detector will start detecting and fetching the output data (detected bounding boxes and labels) to the auto-labeling module. Our auto-labeling module has several functions, the most important feature is the function to write down the information of bounding boxes and labels into existing annotation files or new brand ones. Another one is a function for extracting annotation files. For example, when the existing dataset such as [19] has existing annotation files with several classes (labels), this model can extract annotation files consisting of the "jellyfish" class and edit the existing classes to a new name (i.e., "jellyfish"). Besides, this method can also support searching and downloading videos automatically with desired keywork from YouTube in.mp4" format, and inferencing videos to create annotation files by Youtube_Scapper functions.

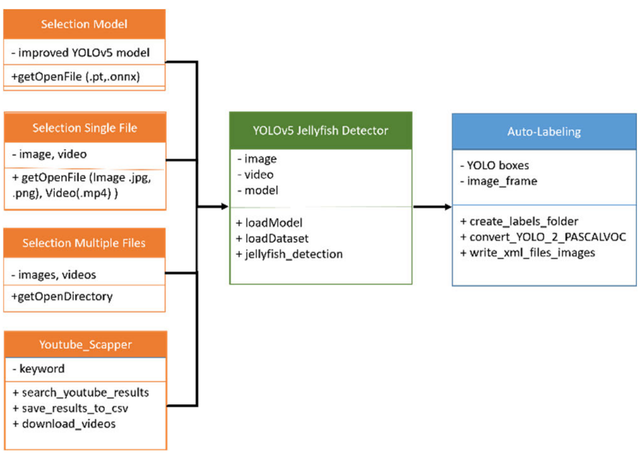

FIGURE 4. UML of our YOLOv5-assisted annotation application.

B. IMPROVED YOLO-BASED MODELS

1) GAM ATTENTION

Applying attention mechanisms to improve the performance of image detection and classification has been widely used in many previous studies. Several dominant methods in the development of attention mechanisms are listed. The first method to combine channel attention and channel-wise feature-fusion to bypass the irrelevant channels is Squeeze-and-Excitation Networks (SENet) [42]. It is less effective, nevertheless, at suppressing irrelevant pixels. The latter attention mechanisms took both channel and spatial factors into account. In detail, while the Bottleneck Attention Module (BAM) [43] performed the channel and spatial operation simultaneously, the Convolutional Block Attention Module (CBAM) did so sequentially.

Nevertheless, as a result of their ignorance of channel-spatial interactions, both attention modules lose the cross-dimension information. In contrast, the TAM took into account the importance of cross-dimension interactions, which increased efficiency by using the attention weights between each pair of the three dimensions, including channel, spatial width, and height. Therefore, TAM is only applied on two of three attention weights, instead of always applying attention operations on all three dimensions. Liu et al. [44] introduced a Global Attention Mechanism (GAM) that can catch important information in all three dimensions to amplify cross-dimension interactions as shown in Fig. 5.

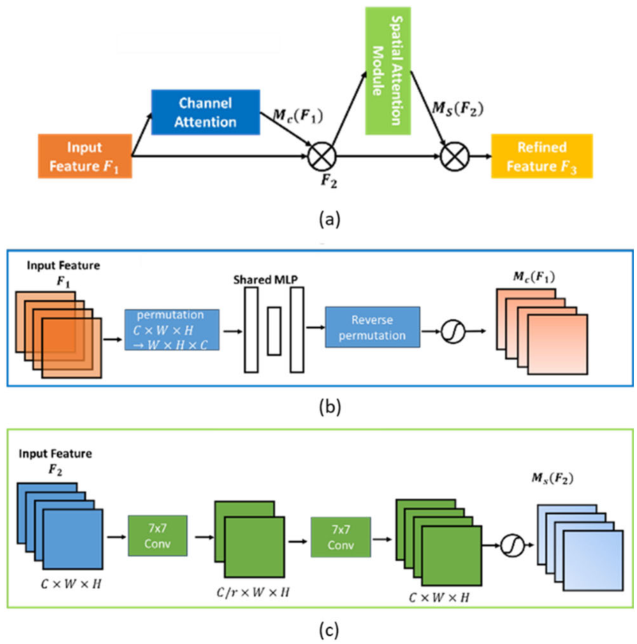

FIGURE 5. Overview of GAM architecture: (a) GAM structure, (b) Channel attention module, and (c) Spatial attention module.

The general methodology is represented by Equations 1 and 2, which includes the input feature map $F_1 \in R^{C \times H \times W}$, the intermediate state F_2, and the output F_3 as follows:

$$F_2 = M_c (F_1) \otimes F_1 \tag{1}$$

$$F_3 = M_S (F_2) \otimes F_2 \tag{2}$$

where M_c and M_S are the channel and spatial attention maps, respectively as shown in (b) and (c) respectively; $\otimes$ denotes element-wise multiplication.

2) COORD_COV

A coordinate convolution (CoordConv) layer performs similar functions as a standard conventional layer, but it achieves the mapping by first concatenating additional channels to the incoming representation. In detail, Fig. 6 presents the comparison of 2D convolutional and Coord_Conv layers [45].

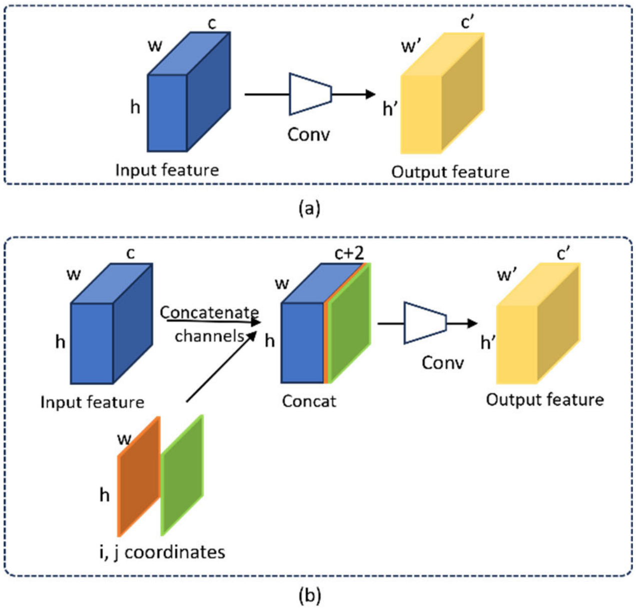

FIGURE 6. Structure of convolutional layer: (a) Standard convolutional layer, (b) Coordinate convolutional layers (CoordConv).

As can be seen in Fig. 6(a), a representation block with the shape of $h \times w \times c$ is mapped to a new form representation with the shape of $h' \times w' \times c'$ by a conventional standard convolutional layer. While in Fig. 6(b), a CoordConv layer firstly concatenates additional channels that hard-code i, j coordinates are contained in these channels. The CoordConv layer preserves the effective computation and low-parameter requirements of a conventional one while letting the network learn whether to retain translation invariance or reject it depending on the learned task. This is helpful for problems involving coordinate transforms when the conventional method might not work. When utilizing CoordConv, a Faster R-CNN detection model trained on MNIST detection demonstrated a 24% better IOU when using CoordConv. In this work, we verify the benefits of using CoordConv on the localization task of jellyfish with better IOU.

IV. EXPERIMENT AND ANALYSIS

A. DATASET

Using the model-assisted annotation method, we created a jellyfish dataset including 14,376 images with corresponding labeled annotations in VOC format. Fig. 7 illustrating some typical jellyfish images in our dataset. Table 2 presents the dataset configuration, including training, validation, and testing sets. Our dataset is utilized in training and validation sets to train SOTA models, which are divided into 80/20 percentages with 11500 images and 2876 images, respectively. The open-source dataset [19] is used as a testing set to verify the generalization of our improved YOLOv5n model and baseline with 842 images and 976 labels.

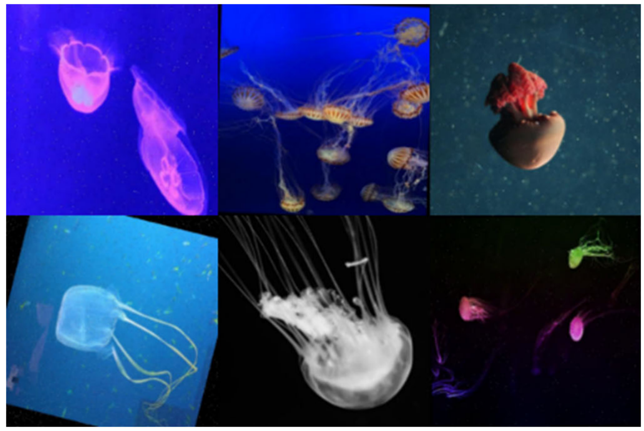

FIGURE 7. Some images are examples of training/evaluation sets.

TABLE 2. Dataset configurations.

Dataset	Division	Images	Labels
Our dataset	Training	11500	10576
	Validation	2876	
Open-source dataset [19]	Testing	842	976

Besides, alternative testing images are extracted from YouTube videos [45] for comparison of SOTA models. Fig. 8 presents the testing images with 15 different species of jellyfish. It is noted that the testing data is not included in the training/validation set. The shape, appearance, color, and size of jellyfish are significant variations according to different types, which presents the diversity of the dataset and enhances the challenges of the detection task. Moreover, the testing images are recorded in two ways: normal cameras underwater, and on the surface of the water.

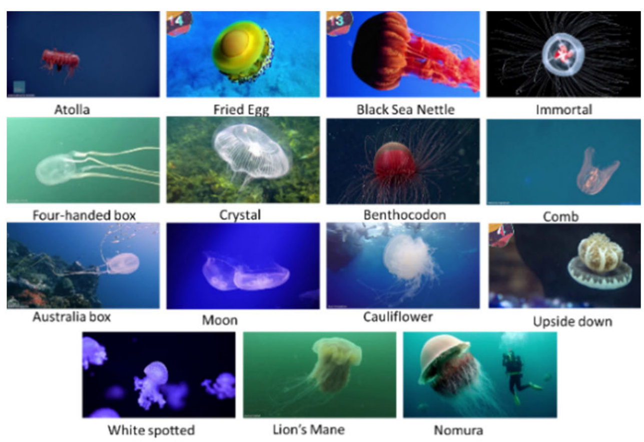

FIGURE 8. Jellyfish species of testing set: 15 species.

In addition, many factors impact images such as ambient noise, the background, the light, and the appearance of other marine species that burden the detection task of models. Thus, apart from YOLOv5 default hyperparameters data augmentation like a mosaic, copy_paste, we apply some image processing techniques (i.e, motion blur, random brightness contrast, CLAHE, To Gray) while training YOLOv5 model using albumentation library [46] as can be seen in Fig. 9.

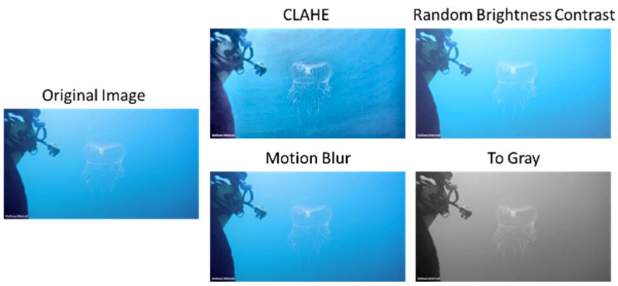

FIGURE 9. **Image pre-processing methods during training YOLOv5.**

It is noted that in the previous work in [47], their dataset images were applied to image pre-processing methods and then used to train models. While in our work, YOLOv5 framework natively supports applying albumentation library while training by customizing the image processing methods and the probability of occurrence in YOLOv5 training options.

B. EXPERIMENTS SETUP

To verify the efficiency of our improved YOLO model, various ablation experiments are conducted. Firstly, to improve the detection rate and bounding boxes localization of the model, we evaluate the feasibility of adding attention mechanisms into the YOLOv5n baseline, including Coordinate Attention (CA), Convolutional Block Attention Module (CBAM), Triplet Attention Module (TAM), Efficient Channel Attention (ECA), Global Attention Mechanism (GAM), and replacing Convolutional layer (Conv) with Coordinate convolutional layer (CoordConv). Besides, we further optimize our improved YOLOv5n model by fine-tuning with pre-trained weight. Secondly, we compare the best model of our work (YOLOv5n-GAM-CoordConv) versus other models including SSD, Retinanet, Faster R-CNN, and YOLO series in the same environment. To fairly evaluation, these models are trained from scratch without using their pre-trained weights and with the same number of 30 epochs and other default settings of their models without any hyperparameter optimization. The model training framework is included Ultralytics for YOLOv5, Meituan for YOLOv6, and open-mmlab (mmdetection) [48] for Faster R-CNN-R50 (Resnet50_FPN) and SSD300, and Retinanet (Resnet50_FPN).

Our environment is equipped with NVIDIA RTX 3050 GPU, CUDA 11.6, and Pytorch 2.0. The precision, recall, average precision (AP), and mean average precision with IoU of 0.5 (mAP@0.5) [49] are used for the evaluation of these models.

C. ABLATION STUDY

The ablation experiments are performed by adding different attention modules and replacing Conv with CoordConv into the YOLOv5n baseline consisting of 7 models: YOLOv5n-CA, YOLOv5n-CBAM, YOLOv5n-ECA, YOLOv5n-TAM, YOLOv5n-CoordConv, YOLOv5n-GAM, YOLOv5n-GAM-CoordConv models. The results of the ablation study are presented in Table 3.

TABLE 3. **Ablation study.**

Models	Paras (M)	FLOPs	mAP @0.5	mAP @.5::95	Precision	Recall
YOLOv5n (baseline)	1.76	4.1	87.8	52	81.4	84.4
w/ CA	1.78	4.2	88	51.5	81.9	85
w/ CBAM	1.78	4.2	88.5	52.6	82.7	84.3
w/ ECA	1.76	4.1	88.2	52.1	82.7	84.4
w/ TAM	1.76	4.3	88.4	52.4	81.4	85.2
w/ CoordCov	2.54	5.6	88.7	53.5	82.3	85
w/ GAM	2.19	4.5	88.8	0.532	83	84.8
w/ GAM-CoordCov	3.55	7.0	89.1	53.3	83.5	84.8
w/ GAM-CoordCov-pre-trained weight	3.55	7.0	89.7	57.6	87.3	85.4

Following the timeline of the attention module release, the parameter and computational cost are higher than the previous. As a result, the complexity of the network and computational cost contribute to the improvement of accuracy and increased precision and recall rate on jellyfish detection. Our work (YOLOv5n-GAM-CoordConv) achieves the highest increase in detection accuracy by 1.3 percent (with @mAP0.5 of 87.8 percent for YOLOv5n baseline and 89.1 percent for the YOLOv5n-GAM-CoordConv). This increased accuracy comes at the trade-off of doubling the network parameters and over 1.5 times the FLOPs computation cost of the baseline model. Achieving a balance between precision and recall is crucial in deep-learning object detection. A minimal gap between precision and recall indicates that the model achieves high-accuracy detection. In this ablation analysis, the YOLOv5-GAM-CoordConv model exhibits the smallest gap between precision and recall, at 1.3 percent. In contrast, the YOLOv5 baseline, YOLOv5n-GAM, and YOLOv5n-CoordConv models display larger gaps, at 3, 1.8, and 2.7 percent, respectively.

Besides, we apply transfer learning using pre-trained weight from a larger source dataset namely COCO dataset to fine-tuning our improved model on custom jellyfish dataset for further optimization. Taking advantage of pre-trained weight, the mAP@0.5 accuracy of our improved model increases 0.6 percent from 89.1 percent in case of training-from-scratch to 89.7 percent. Other evaluation metrics including precision, recall, and mAP@0.5:0.95 also experience the boost, which verifies the effectiveness of fine-tuning with pre-trained weight using transfer learning.

To validate our improved model generalization ability, we perform testing on an open-source dataset with 842 images in case of training from scratch and using pre-trained weights. Fig. 10 presents the evaluation results, which verify the generalization ability of our improved model. While training-from-scratch, our improved model gets the inference result mAP@0.5 accuracy at 66.9 percent on 842 testing images, which is quite a promising result for inferencing new images. By leveraging pre-trained weights, it boosts the mAP@0.5 accuracy by 7.9 percent. By fine-tuning the model with pre-trained weight, our improved model is better adapted to specific features and patterns in our custom jellyfish dataset, leading to improvement of performance and generalization ability.

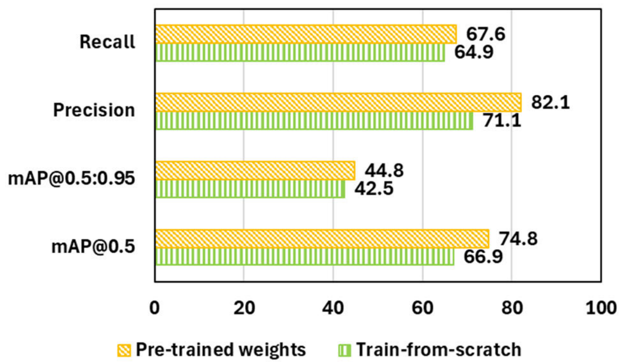

FIGURE 10. Evaluation results on a testing set of 842 images of our improved models in case of training-from-scratch and pre-trained weights.

D. ANALYSIS

In this work, we compare our work (improved YOLOv5n-GAM-CoordConv) with other models, which are frequently used in object detection tasks. Together with the YOLOv6n, YOLOv8n, SSD, and Retina are the most famous one-stage detectors; and Faster R-CNN is the most well-known two-stage detector.

As shown in Table 4, apart from the YOLOv5n baseline, our work has the smallest computation cost FLOPs at 7.0 with the highest accuracy in terms of mAP@0.5 at 89.1 percent. Compared to the YOLOv5n-baseline, our work solves the problem of low precision and even increases the recall rate. From 81.4 up to 83.5 percent for Precision, and from 84.4 reaches to 84.8 percent for Recall. Low precision in object detection refers to a situation where the model identifies many objects, but a significant portion of those identifications are incorrect. A higher recall rate in object detection indicates that the model is effectively finding a larger proportion of the actual objects present in the images. The quantitative values of precision and recall indicate that our improved YOLOv5n-GAM-CoordConv model feasibly detects jellyfish precisely without over-detection with errors (low precision) and miss detection of the correct jellyfish (high recall). Besides, F1-score is the combination of precision and recall, which also states the outperform of our improved model compared to other models with 84 and 83 percent respectively. It is noted that F1-score is not reported by native openmmlab training platform, so we note it as "./."

TABLE 4. Models comparison.

Models	Paras (M)	FLOPs	mAP @0.5	Precision	Recall	F1-score
SSD300	34.31	154.5	60	86.4	67.4	./.
Retina (Resnet50-FPN)	37.7	95.7	67.8	78.8	75	./.
Faster R-CNN (Resnet50-FPN)	41.5	91.4	72.4	54.4	81.5	./.
YOLOv5n (baseline)	1.76	4.1	87.8	81.4	84.4	83
YOLOv6n	4.3	11.06	87.7	83.8	81.6	83
YOLOv8n	3.0	8.2	88.7	84.4	81.7	83
Our work	3.55	7.0	89.1	83.5	84.8	84

The visual qualitative results on the testing images also verify this indication. As can be seen in Fig. 11, the testing images are very challenging generalization verification, including the small resolution of the whole jellyfish (a), the top-down of jellyfish with many stings (b), only a big part of jellyfish showing in the image (c), a jellyfish with long tails (d), a crystal jellyfish and human background (e), a jellyfish with long tails with water bubble background (f), a jellyfish with long tails and other small fishes background (g), another jellyfish with long tails and other small fishes background (h), a mushroom shape jellyfish with light reflection background (i), a jellyfish with a similar color to ambient background (j), an upside-down jellyfish with light reflection background (k), many jellyfish overlapping with each other and bubble ambient background (l), a jellyfish on the surface of water (m).

It can be seen that our work overperforms other models in the case of these testing images. YOLOv6 and YOLOv8 prove to be very good jellyfish detectors, only failing behind our work in the case of images: (b), (e), (l), and (m) for YOLOv6; images (d), (e), (f), (i), and (m) for YOLOv8. Especially, in the case of image (e), the color of the jellyfish is crystal and the human diving equipment looks like jellyfish making it very difficult to distinguish correctly. And our work can detect precisely without wrong over-detection.

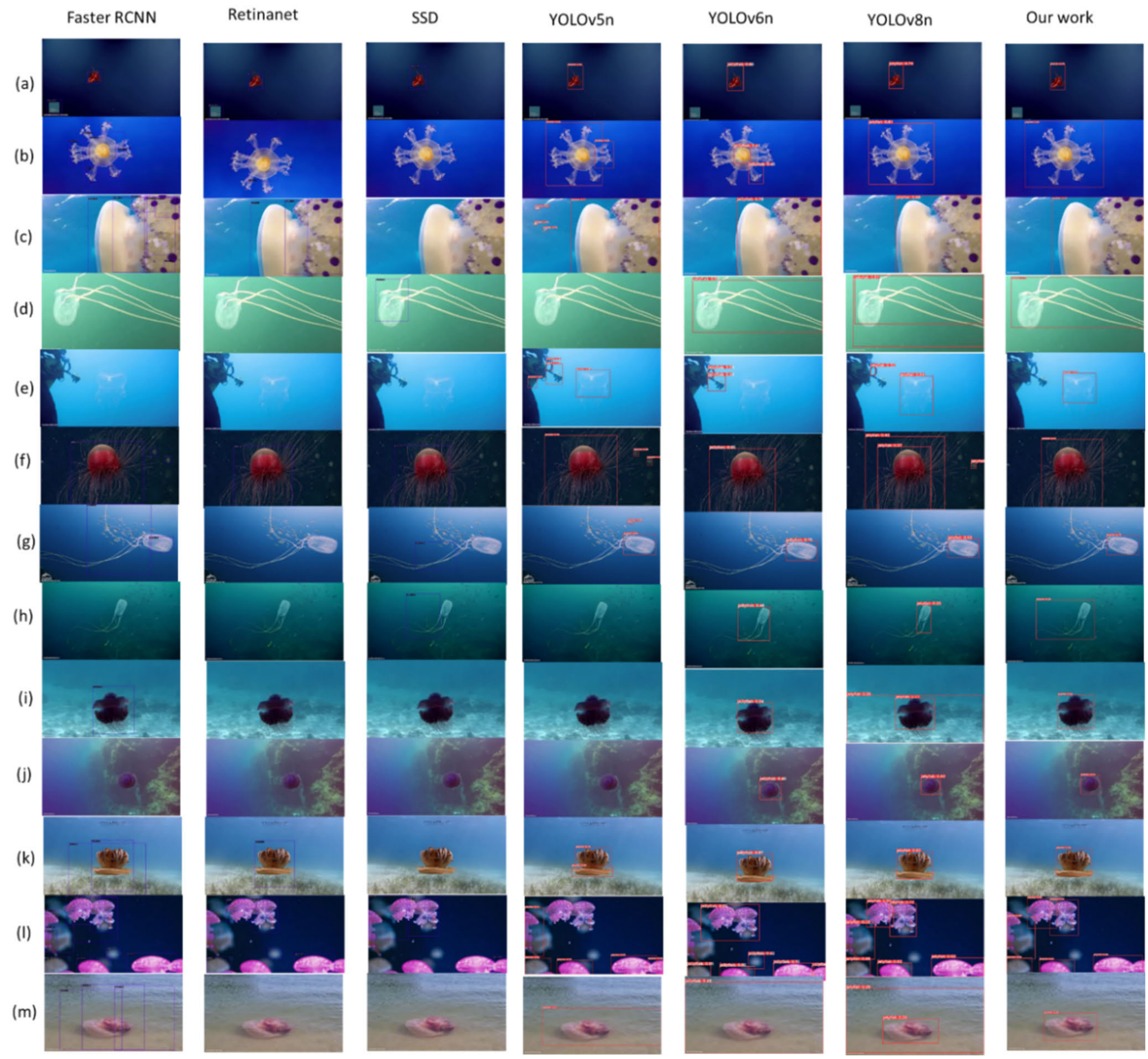

FIGURE 11. **Qualitative Visualization Results of Different Models.**

In the case of the overlapping of jellyfish in the image (l) and the jellyfish captured on the surface of the water in image (m), our work proves the benefits of utilizing GAM attention to correctly detect jellyfish. However, there is still some room for improvement. For example, our dataset consists only of the underwater capturing of jellyfish, which can be extended to the approach of jellyfish captured by drones or UAVs. Following this direction, we can develop the jellyfish bloom detector using deep learning with images from drones or UAVs.

In this work, we focus on improving the detection rate and the accuracy of localization of some underwater jellyfish. In the case of detecting jellyfish on the surface of water or captured by drones, the small resolution of jellyfish versus the big scene of sea background, the ambient noise, and the lightning reflection will be different problems that need to be considered. Our comprehensive system is deployed into the application that can be easily customized into other categories, i.e., fishes, marine species, and marine waste [50], [51], [52]. Our work specializes in underwater imaging which can be extended to UAV-capturing images for jellyfish distribution (or jellyfish bloom) detection [53]. Besides, the deployment of a deep learning-based object detection system on embedded devices (Nvidia Jetson devices) on USVs and UAVs also will be a considerable approach. In our previous study in [54], we experienced the deployment on Nvidia Jetson Nano and Xavier, which provided some insight into our further work on jellyfish project. It is worth considering that Ultralytics, the YOLOv5 and YOLOv8 development platform, fully supports the exportation of training weight from native Pytorch training platform to TensorRT to boost inference time on Nvidia Jetson devices. However, the mismatch of Pytorch, ONNX, and TensorRT versions between the training GPU device RTX3050 and Nvidia Jetson devices may cause exportability problems. Another issue of deployment is resource constraints in limited computational resources on embedded devices. Compared to YOLOv8, the latter version released by Ultralytics, our improved YOLOv5n model has less computation complexity, which is a good point. However, adding an attention module into default source code needs further optimization while exporting to TensorRT and deploying into Nvidia Jetson devices. With the release of Orin [55] the new generation of Nvidia Jetson devices, compared to our work with older generation devices, it is quite promising to achieve nearly real-time performance.

V. CONCLUSION

This paper presents a YOLO-based deep learning model assisted-annotation system for real-time jellyfish detection from underwater video/image recordings. Our system deploys a YOLOv5n pre-trained model to detect jellyfish and auto-labeling to generate annotations automatically based on YOLOv5-detected bounding boxes and labels, which can save a lot of manual time and labor.

To increase the auto-labeling effectiveness, it is essential to improve the accuracy of detection and localization of jellyfish. Therefore, we improve the YOLOv5n baseline network structure by adding a GAM module and replacing conventional Conv with CoordConv to maintain fast detection time and enhance generalization performance. From the quantitative experiments results, our improved YOLOv5n model achieves the highest accuracy of mAP@0.5 at 89.1%, which outperforms other SOTA models, including Faster RCNN, YOLOv6, and YOLOv8. The qualitative testing experiments also verify these indications, as our models can accurately detect jellyfish in many difficult cases, including small, occupied, and big portions of jellyfish, and noisy ambient backgrounds with fish, human, and diving equipment which surpass the performance of other models. Further work will focus on the deployment of our system on embedded devices (i.e., Nvidia Jetson) as well as the customization of our application back-end code to YOLOv6, and YOLOv8.

VI. AVAILABILITY OF DATA AND MATERIALS

Please contact the corresponding author for data requests. We use Visual Studio Code as a coding IDE and Jupyter Notebook as a training IDE, which includes support for Python programming. Our model-assisted annotation application is developed with Python language as the back-end and PyQT5 as the front-end GUI. We provide the demo video of our model-assisted annotation application in a GitHub repository [56] together with training experiment results, model weights, and testing image results in a GitHub repository [57].

REFERENCES

[1] L. Yan, S. Li, and F. Ding, "The preliminary studies on the dynamics of macro-jellyfish resources and their relationship with fisheries in the East China Sea and yellow sea," *Mar. Fisheries*, vol. 1, pp. 10–14, Aug. 2004.

[2] J. Dong et al., "The morphology and structure of jellyfish (Cyanea nozakii kishinouye)," *Fish Sci.*, vol. 24, no. 2, pp. 22–23, 2005.

[3] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proc. IEEE*, vol. 111, no. 3, pp. 257–276, Mar. 2023.

[4] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and efficient foundation language models," 2023, *arXiv:2302.13971*.

[5] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.

[6] I. de Zarzà, J. de Curtò, G. Roig, and C. T. Calafate, "LLM multimodal traffic accident forecasting," *Sensors*, vol. 23, no. 22, p. 9225, Nov. 2023, doi: 10.3390/s23229225.

[7] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," 2023, *arXiv:2304.08485*.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[9] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1440–1448.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[12] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. 14th Eur. Conf.* Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 21–37.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.

[14] B. Alexey, C.-Y. Wang, and H.-Y. Liao, "YOLOV4: Optimal speed and accuracy of object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Apr. 2020, pp. 1–17.

[15] G. Jocher. (2020). *YOLOV5*. [Online]. Available: https://github.com/ultralytics/yolov5

[16] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, and X. Wei, "YOLOV6: A single-stage object detection framework for industrial applications," 2022, *arXiv:2209.02976*.

[17] (2023). *YOLOV8*. [Online]. Available: https://github.com/ultralytics/ultralytics

[18] J. Redmon and A. Farhadi, "YOLOV3: An incremental improvement," 2018, *arXiv:1804.02767*.

[19] M. Martin-Abadal. (2019). *Jellyfish Object Detection*. Accessed: Mar. 17, 2020. [Online]. Available: https://github.com/srv/jf_object_detection

[20] T.-N. Pham, V.-H. Nguyen, and J.-H. Huh, "Integration of improved YOLOV5 for face mask detector and auto-labeling to generate dataset for fighting against COVID-19," *J. Supercomput.*, vol. 79, no. 8, pp. 8966–8992, May 2023.

[21] H. Kim, D. Kim, S. Jung, J. Koo, J.-U. Shin, and H. Myung, "Development of a UAV-type jellyfish monitoring system using deep learning," in *Proc. 12th Int. Conf. Ubiquitous Robots Ambient Intell. (URAI)*, Oct. 2015, pp. 495–497.

[22] J. Koo, S. Jung, and H. Myung, "A jellyfish distribution management system using an unmanned aerial vehicle and unmanned surface vehicles," *IEEE Underwater Technol. (UT)*, vol. 2017, no. 1, pp. 1–5, Jun. 2017.

[23] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.
[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.
[26] G. French, M. Mackiewicz, M. Fisher, M. Challiss, P. Knight, B. Robinson, and A. Bloomfield, "JellyMonitor: Automated detection of jellyfish in sonar images using neural networks," in *Proc. 14th IEEE Int. Conf. Signal Process. (ICSP)*, Aug. 2018, pp. 406–412.
[27] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *Proc. 3rd IAPR Asian Conf. Pattern Recognit. (ACPR)*, Nov. 2015, pp. 730–734.
[28] M. Martin-Abadal, A. Ruiz-Frau, H. Hinz, and Y. Gonzalez-Cid, "Jellytoring: Real-time jellyfish monitoring based on deep learning object detection," *Sensors*, vol. 20, no. 6, p. 1708, Mar. 2020.
[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-V4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, Feb. 2017, pp. 2354–2362.
[30] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 2951–2962, Dec. 2016.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[32] Y. Han, Q. Chang, S. Ding, M. Gao, B. Zhang, and S. Li, "Research on multiple jellyfish classification and detection based on deep learning," *Multimedia Tools Appl.*, vol. 81, no. 14, pp. 19429–19444, Jun. 2022.
[33] C. Qiuyue, "Research on jellyfish detection algorithm based on convolutional neural network," M.S. thesis, Yanshan Univ., Qinhuangdao, China, 2021.
[34] M. Gao, Y. Bai, Z. Li, S. Li, B. Zhang, and Q. Chang, "Real-time jellyfish classification and detection based on improved YOLOV3 algorithm," *Sensors*, vol. 21, no. 23, p. 8160, Dec. 2021.
[35] M. Gao, S. Li, K. Wang, Y. Bai, Y. Ding, B. Zhang, N. Guan, and P. Wang, "Real-time jellyfish classification and detection algorithm based on improved YOLOV4-tiny and improved underwater image enhancement algorithm," *Sci. Rep.*, vol. 13, no. 1, pp. 1–16, Aug. 2023.
[36] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Computer Vision–ECCV*. Cham, Switzerland: Springer, 2018, pp. 3–19.
[37] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nashville, TN, USA, Jun. 2021, pp. 13708–13717.
[38] D. Misra, T. Nalamada, A. U. Arasanipalai, and Q. Hou, "Rotate to attend: Convolutional triplet attention module," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 3138–3147.
[39] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient channel attention for deep convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 11531–11539, doi: 10.1109/CVPR42600.2020.01155.
[40] Y. Liu, Z. Shao, and N. Hoffmann, "Global attention mechanism: Retain information to enhance channel-spatial interactions," 2021, *arXiv:2112.05561*.
[41] [Online]. Available: https://github.com/tzutalin/labelImg/
[42] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
[43] J. Park, S. Woo, J.-Y. Lee, and I. So Kweon, "BAM: Bottleneck attention module," 2018, *arXiv:1807.06514*.
[44] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the CoordConv solution," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–12.
[45] *Top Fives, 15 Incredible Jellyfish Species*. [Online]. Available: https://www.youtube.com/watch?v=8_9toXCB4sU
[46] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," 2018, *arXiv:1809.06839*.
[47] V. Lopez-Vazquez, J. M. Lopez-Guede, D. Chatzievangelou, and J. Aguzzi, "Deep learning based deep-sea automatic image enhancement and animal species classification," *J. Big Data*, vol. 10, no. 1, p. 37, Mar. 2023.
[48] [Online]. Available: https://github.com/open-mmlab/mmdetection
[49] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision–ECCV*. Springer, 2014, pp. 740–755.
[50] G. Yu, R. Cai, J. Su, M. Hou, and R. Deng, "U-YOLOV7: A network for underwater organism detection," *Ecol. Inform.*, vol. 75, Jul. 2023, Art. no. 102108.
[51] S. R and M. M, "Comparing YOLOV3, YOLOV5 & YOLOV7 architectures for underwater marine creatures detection," in *Proc. Int. Conf. Comput. Intell. Knowl. Economy (ICCIKE)*, Mar. 2023, pp. 25–30.
[52] I. de Zarzà, J. de Curtò, and C. T. Calafate, "Detection of glaucoma using three-stage training with efficientNet," *Intell. Syst. Appl.*, vol. 16, Nov. 2022, Art. no. 200140.
[53] W. Zhang, F. Rui, C. Xiao, H. Li, and Y. Li, "JF-YOLO: The jellyfish Bloom detector based on deep learning," *Multimedia Tools Appl.*, vol. 83, no. 3, pp. 7097–7117, Jan. 2024.
[54] H.-V. Nguyen, J.-H. Bae, Y.-E. Lee, H.-S. Lee, and K.-R. Kwon, "Comparison of pre-trained YOLO models on steel surface defects detector based on transfer learning with GPU-based embedded devices," *Sensors*, vol. 22, no. 24, p. 9926, Dec. 2022.
[55] *Nvidia Jetson Orin*. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/
[56] N.-T. Pham, H.-V. Nguyen, and J.-H. Huh. [Online]. Available: https://github.com/MiuMiao93/Integration-system-of-improved-YOLOv5-face-mask-detection-and-auto-label-module
[57] N.-T. Pham, H.-V. Nguyen, K.-R. Kwon, J.-H. Kim, and J.-H. Huh. [Online]. Available: https://github.com/MiuMiao93/Integration-system-of-improved-YOLOv5-face-mask-detection-and-auto-label-modu

THI-NGOT PHAM received the bachelor's degree in engineering physics from Vietnam National University (VNU) University of Engineering and Technology, Hanoi, Vietnam, in 2015, and the master's (M.E.) degree from the Department of Biomedical and Engineering, Pukyong National University, Busan, Republic of Korea, in 2018. She is currently pursuing the Ph.D. degree with the Department of Data Informatics, National Korea Maritime and Ocean University, Busan.

She was a full-time Researcher with the Big Data Center for Total Lifecycle of Shipbuilding and Shipping, National Korea Maritime and Ocean University, from December 2020 to August 2021. Since September 2022, she has been with the Join Global R&E Program for Interdisciplinary Technologies of Ocean Renewable Energy (BK 21 Four Research Group), Interdisciplinary Major of Ocean Renewable Energy Engineering, National Korea Maritime and Ocean University.

VIET-HOAN NGUYEN received the bachelor's degree in electronic and telecommunication from Hanoi University of Science and Technology, Vietnam, in 2014, and the master's degree from the Department of IT Convergence and Application Engineering, Pukyong National University, Busan, Republic of Korea, in 2017. He is currently pursuing the Ph.D. degree with the Department of Artificial Intelligence, Pukyong National University, Daeyeon Campus.

Since November 2017, he has been with the Research and Development Center, Intown Company, Busan. He received the Best Paper Award at the Academic Conference, Busan, in 2016.

KI-RYONG KWON (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronics engineering from Kyungpook National University, in 1986, 1990, and 1994, respectively.

He was with Hyundai Motor Company, from 1986 to 1988. He was with Pusan University of Foreign Language, from 1996 to 2006. Since 2006, he has been a Full Professor with the Department of IT Convergence and Application Engineering, Pukyong National University, Republic of Korea. He was the Dean of the College of Engineering, Pukyong National University, from September 2018 to September 2020. He was a Postdoctoral Researcher with the University of Minnesota, USA, from 2000 to 2002. He was a Visiting Professor with Colorado State University, from 2011 to 2012.

Dr. Kwon received a commendation at Japan NOLTA Society's Complex Systems Communication (CCS) Research Symposium, in January 2018. Additionally, he was a recipient of the 15th Busan Science and Technology Award, in 2016. He was the Organizing Chair of the Second International Conference on Multimedia Information Technology and Applications (MITA 2006) and the 12th International Conference on Multimedia Information Technology and Applications (MITA 2016: Souphanouvong University at Laos). He was the General Chair of the 11th International Conference on Multimedia Information Technology and Applications (MITA 2015: Uzbekistan Presidential Palace at Toshkent). He was the President of Korea Multimedia Society, from January 2015 to December 2016. He was also the Director of the IEEE R10 Changwon Section, Republic of Korea, from March 1996 to February 2006. He was the Editor-in-Chief (EiC) of the *Journal of Korea Multimedia Society*, Korea Multimedia Society (KCI-indexed), from January 2006 to December 2007.

JAE-HWAN KIM received the B.S. degree in industrial engineering from Korea University, Seoul, Republic of Korea, in February 1984, the M.S. degree from the Department of Industrial and Systems Engineering, Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea, in February 1987, and the Ph.D. degree from the Department of Industrial and Systems Engineering, Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea, in February 1993.

He was a Postdoctoral Researcher with the University of Illinois, USA (Supported by KOSEF), from September 1997 to August 1998. He was a Visiting Scholar of industrial and systems engineering with the Samuel Ginn College of Engineering, Auburn University, USA, from September 2011 to August 2012. He has been a Full Professor (Tenured) with the Department of Data Science, National Korea Maritime and Ocean University, Republic of Korea, since 1991 September.

Dr. Kim has received numerous outstanding paper awards. He delivered a keynote speech on "A Hybrid Metaheuristic for the Series-Parallel Redundancy Allocation Problem" at the 2008 KMU-HCMUT Joint Workshop and also gave tutorial lectures at the 2016 Korean Data and Information Science Society Fall Conference. After being published in IEEE TRANSACTIONS ON RELIABILITY journal, in 1993, he has served as a reviewer for IEEE TRANSACTIONS for over 30 years.

JUN-HO HUH (Member, IEEE) received the B.S. degree in science from the Department of Major of Applied Marine Sciences, the B.E. degree in engineering (double major) from the Department of Computer Engineering, Jeju National University at Ara, Jeju, Republic of Korea, in August 2007, the M.A. degree in education from the Department of Computer Science Education, Pukyoug National University, Daejeon, Busan, Republic of Korea, in August 2012, and the Ph.D. degree in engineering from the Department of Computer Engineering, Graduate School, Pukyoug National University, in February 2016.

He finished the Cooperative Marine Science and Engineering Program, Texas A&M University at Galveston, USA, in August 2006. He was a Research Professor with Dankook University, Jukjeon, Yongin, Republic of Korea, from July 2016 to September 2016. He was an Assistant Professor with the Department of Software, Catholic University of Pusan, Republic of Korea, from December 2016 to August 2019. He was an Assistant Professor with the Department of Data Informatics, National Korea Maritime and Ocean University, Busan, from September 2019 to September 2021. Since October 2021, he has been an Associate Professor (Tenured) with the Department of Data Informatics/Data Science, National Korea Maritime and Ocean University. Since September 2020, he has been the Center Chair (Director) of the Big Data Center for Total Lifecycle of Shipbuilding and Shipping, National Korea Maritime and Ocean University. Since September 2022, he has been a Joint Associate Professor (Tenured) with the Global R&E Program for Interdisciplinary Technologies of Ocean Renewable Energy (BK 21 Four Research Group), Interdisciplinary Major of Ocean Renewable Energy Engineering, National Korea Maritime and Ocean University. He was the General/Head Professor with the Catholic University of Pusan on International Game Exhibition G-Star 2017 (G-Star 2017). He is the author of the book *Smart Grid Test Bed Using OPNET and Power Line Communication* (USA: IGI Global, 2017, 425) and *Principles, Policies, and Applications of Kotlin Programming* (USA: IGI Global, 2023, 457). He has authored/edited ten books and edited ten special issues in reputed Clarivate Analytics Index journals. He has published more than 100 articles in the Clarivate Analytics Index (SCI/SCIE/SSCI indexed) with over 3200 citations and has an H-index of 32.

Dr. Huh has been a Technical Committee (TC) Member of the International Federation of Automatic Control (IFAC), CC 1 (Systems and Signals), TC 1.5. (Networked Systems); IFAC, Computers, Cognition and Communication (CC 3), TC 3.2 (Computational Intelligence in Control); and IFAC, CC 7 (Transportation and Vehicle Systems), TC 7.2. (Marine Systems), since 2017. Since 2020, he has been a TC Member of IFAC, CC 2 (Design Methods), TC 2.6. (Distributed Parameter Systems). He received the Best Paper Minister Award (Ministry of Trade, Industry and Energy, Korean Government) from the 16th International Conference on Control, Automation, and Systems, in October 2016, ICROS with IEEE Xplore, the Springer Nature Most Cited Paper Award, and the *Human-centric Computing and Information Sciences* Most Cited Paper Award, in 2019 (Research published in the journal, from 2016 to 2018; SCIE IF=6.558). He was awarded the Commendation for Meritorious Service in the Promotion of Busan's Data Industry, in December 2023 (Commendation by the Mayor of Busan). He was the Organizing Chair of the 15th International Conference on Multimedia Information Technology and Applications (MITA 2019: University of Economics and Law (UEL), Vietnam National University Ho Chi Minh City, Vietnam) and the 17th International Conference on Multimedia Information Technology and Applications (MITA 2021: Jeju KAL Hotel, ROK). He was the Managing Editor (ME) of the *Journal of Information Processing Systems* (JIPS), Korea Information Processing Society (SCOPUS/ESCI indexed), from January 2020 to December 2021, and the *Journal of Multimedia Information System* (JMIS), Korea Multimedia Society (EI/KCI indexed), from January 2017 to December 2022. Since July 2023, he has been the Editor-in-Chief (EiC) of *Global Convergence Research*, Global Convergence Research Academy, Pukyoug National University. He is an Associate Editor (AE) of *Journal of Information Processing Systems* (JIPS), Korea Information Processing Society (SCOPUS/ESCI indexed); the *Journal of Multimedia Information System* (JMIS), Korea Multimedia Society (EI/KCI indexed); and *Human-centric Computing and Information Sciences* (HCIS), Springer Berlin Heidelberg (SCIE IF=6.558). For more information, see https://scholar.google.co.kr/citations?user=cr5wjNYAAAAJ&hl.