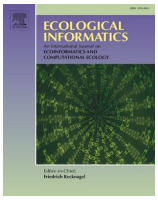
Ecological Informatics 72 (2022) 101847

Contents lists available at ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf



# YOLO-Fish: A robust fish detection model to detect fish in realistic underwater environment

Abdullah Al Muksit [a], Fakhrul Hasan [a], Md. Fahad Hasan Bhuiyan Emon [a], Md Rakibul Haque [b], Arif Reza Anwary [c], Swakkhar Shatabda [a,*]

[a] Department of Computer Science and Engineering, United International University, Dhaka 1212, Bangladesh
[b] Department of Computer Science and Engineering, Rajshahi University of Engineering and Technology, Rajshahi 6204, Bangladesh
[c] Edinburgh Napier University, Edinburg, Scotland, United Kingdom

ARTICLE INFO

*Keywords:*
Fish detection
Underwater ecosystem
Deep Learning
Object Detection
Dataset

ABSTRACT

Over the last few years, several research works have been performed to monitor fish in the underwater environment aimed for marine research, understanding ocean geography, and primarily for sustainable fisheries. Automating fish identification is very helpful, considering the time and cost of the manual process. However, it can be challenging to differentiate fish from the seabed and fish types from each other due to environmental challenges like low illumination, complex background, high variation in luminosity, free movement of fish, and high diversity of fish species. In this paper, we propose YOLO-Fish, a deep learning based fish detection model. We have proposed two models, YOLO-Fish-1 and YOLO-Fish-2. YOLO-Fish-1 enhances YOLOv3 by fixing the issue of upsampling step sizes of to reduce the misdetection of tiny fish. YOLO-Fish-2 further improves the model by adding Spatial Pyramid Pooling to the first model to add the capability to detect fish appearance in those dynamic environments. To test the models, we introduce two datasets: DeepFish and OzFish. The DeepFish dataset contains around 15k bounding box annotations across 4505 images, where images belong to 20 different fish habitats. The OzFish is another dataset comprised of about 43k bounding box annotations of wide varieties of fish across around 1800 images. YOLO-Fish1 and YOLO-Fish2 achieved average precision of 76.56% and 75.70%, respectively for fish detection in unconstrained real-world marine environments, which is significantly better than YOLOv3. Both of these models are lightweight compared to recent versions of YOLO like YOLOv4, yet the performances are very similar.

## 1. Introduction

Today, underwater fish detection is in high demand for different purposes, such as research in marine science and oceanography and monitoring aquaculture for sustainable fisheries. In 2018, total global capture of fisheries production reached the highest level ever recorded at 96.4 million tonnes. Most of the production comes from captured marine fisheries, 84.4 million tonnes (Fao, 2020). Real-time monitoring of those commercial fisheries will help produce more and match human consumption demand. Underwater videos and images offer a non-intrusive, cost-effective way to collect large volumes of visual data to process the information. However, manual processing of underwater videos and images is labor-intensive, time-consuming, expensive, and prone to fatigue errors. Therefore, the automatic processing of underwater videos for fish detection is an attractive alternative. However, the unrestricted environmental factors such as complex background, luminosity, camouflage foreground, crowded and dynamic background, and so on make the task challenging. These influences compromise the accuracy of fish detection (Salman et al., 2020). Besides that, traditional automatic approaches can not detect underwater fish with reasonable detection rates due to the illumination changes (Li and Cao, 2020).

The arrival of deep learning is a breakthrough for object detection to localize the object with various classes (Szegedy et al., 2013; Zhao et al., 2019). Several pieces of research on underwater fish detection have been conducted using deep learning techniques for different purposes in the last couple of years. Li et al. (2015) used a Fast-RCNN network to deploy an automatic fish identification system. The model performed well with a mean average precision (mAP) of 81.4% and detected 80 times faster than the previous R-CNN on a single fish image. The dataset they used is ImageCLEF collected from Fish4Knowledge (Fisher et al.,

* Corresponding author.
*E-mail address:* swakkhar@cse.uiu.ac.bd (S. Shatabda).

https://doi.org/10.1016/j.ecoinf.2022.101847
Received 9 April 2022; Received in revised form 2 October 2022; Accepted 2 October 2022
Available online 7 October 2022

4916), which has many fish images of different types of species with several appearances. In consecutive research, Li et al. (2016) used this dataset again to train on Faster-RCNN, which got an mAP of 82.7%, and then built another lightweight neural network to improve the detection ability than previous models (Li et al., 2017).

These consecutive research efforts have been put into developing systems to understand the complex underwater environment and distinguish fish objects based on the publicly available Fish4knowledge dataset. However, there are many things to avoid in the Fish4knowledge dataset to train on. For example, it has cropped single fish centered into the frame where visual understanding of the underwater environment will be extensively tricky for a model. Besides the resolution is too low, whereas higher resolutions are important for optimizing deep learning models (Horwath et al., 2020; Sabottke and Spieler, 2020). Other automatic fish detection systems are built on some more popular publicly available datasets (Cutter et al., 2015; Anantharajah et al., 2014), which are also unsuitable for training. For example, the Rockfish (Cutter et al., 2015) and QUT fish dataset (Anantharajah et al., 2014) have the cropping issue the same as Fish4knowledge (Fisher et al., 4916), and they have small number of images. Though QUT fish dataset already used a few fish detection tasks, but it cannot be tested for underwater detection because the image background is removed for most of them (Nour Eldeen et al., 2018; Adiwinata et al., 2020). All of these available datasets do not fully capture the variability and complexity of real-world underwater habitats, which often have the same water conditions, high similarity between the appearance between fish and the elements in the background. Summarizing all of these things shows us that there are many limitations full of discretion on the publicly available fish datasets.

Some of the underwater fish detection work has been done on self-built datasets and reported to be trained by state-of-the-art models achieving good accuracy (Cai et al., 2020; Wang et al., 2021). However, in most cases, the models can fit that particular habitat but not be sustainable in another type of fish in a different kind of habitat. Also, the clarity and illumination of the water vary from place to place and add to the complexity.

In this paper, we introduced two large publicly available datasets consisting entirely of real-world images with full HD resolution for fish detection problem. Earlier, they were only available for fish classification problems. The first one is DeepFish (Saleh et al., 2020) which captured natural underwater images from 20 different real-world environments of Australian marine habitats. The second extensive dataset is OzFish (Australian Institute Of Marine Science, 2020) which is also captured images from Australian marine waters. The illumination of waters varies here a little, but there are many fish species in a single frame, which is more numerous than DeepFish. We have also introduced two deep learning based detection models YOLO-Fish-1 and YOLO-Fish-2, enhanced over the YOLOv3 to handle the uneven complex environment more precisely. YOLO-Fish-1 was developed by optimizing upsample step size to reduce the rate of omitted tiny fish during detection. On the other hand, in YOLO-Fish-2, we add Spatial Pyramid Pooling(SPP) module along with YOLO-Fish-1 to reduce mis-detections overall for complex environments. The experiments were performed on the two datasets proposed. Experimental analysis shows the significance of the proposed models over YOLOv3 and also comparable performances to later versions of YOLO.

## 2. Materials and methods

In this section, we introduce the two newly proposed public datasets for fish detection. We have developed YOLO-Fish imprvosing on the YOLOv3. The rest of the section presents are details of these deep learning based detection models.

### 2.1. Datasets

To add robustness to the model, we merged two datasets that have large-scale collection of fish images. They are: DeepFish (Saleh et al., 2020) and OzFish (Australian Institute Of Marine Science, 2020). DeepFish dataset consists of 20 different habitats acquired from video footage underwater, which holds a substantial variety of an underwater environment. There are three subcategories in the dataset FishClf for classification purposes, localization FishLoc, and semantic segmentation FishSeg (Saleh et al., 2020). Since the DeepFish dataset was not used for detection purposes, so no bounding box annotation was there in the dataset. Thus we selected a sequence of images of different types of movement and posture from every habitat and finally selected 4505 positive images from FishClf. Besides, arround 50% of negative images from each habitat were selected from the same FishClf subcategory where the model grows no interest in those images as it contains no fish. From 4505 image samples, a total of 15463 ground truth bounding boxes were annotated manually using labelImg ( https://github.com/tzutalin/labelImg) tool in YOLO format. A summary of the DeepFish dataset is given in Table 1. Sample images from DeepFish dataset with labels of each habitat is shown in Fig. 1, illustrating the diversity between the habitats.

The OzFish dataset has been developed as part of the Australian Research Data Commons Data Discoveries program to advance research of machine learning for the automated detection of fish from video (Australian Institute Of Marine Science, 2020). The dataset is comprised of around 43k bounding box annotations of fish across almost 1800 frames. The specialty of OzFish dataset is that each picture consists of different species and shapes of many fish instances. In OzFish, each captured frame consists of 25 fish objects on average. Many frames contain up to 80–120 fish objects, where frames of the DeepFish dataset hold 3–4 fish objects on average, and a few frames consist of up to 14 fish objects. Most frames of OzFish consists of many tiny fish objects, and those fish are tough to identify because of their small size. These may belong to the smallest fish species in the sea or minnows. These fish often look like black dots when they are far apart inside the frame. However, we did not omit any single instance of those tiny fish. A few image samples with labels are shown in Fig. 2. The datasets are divided into 80% training set and 20% test with a random stratified selection from each habitat.

**Table 1**
DeepFish dataset overview: number of images annotated and the associated bounding boxes for each habitat.

| Habitat | Annotated images | Bounding boxes |
|---|---|---|
| Rocky mangrove prop roots | 407 | 2995 |
| Sparse algal bed | 462 | 4146 |
| Upper-Mangroave medium Rhizophora | 140 | 140 |
| Sandy mangrove prop roots | 250 | 555 |
| Complex reef | 283 | 611 |
| Low algal bed | 450 | 2031 |
| Seagrass bed | 502 | 1716 |
| Low complexity reef | 300 | 651 |
| Boulders | 149 | 454 |
| Mixed substratum mangrove prop roots | 100 | 198 |
| Reef trench | 207 | 382 |
| Upper mangrove tall rhizophora | 99 | 99 |
| Large boulder | 102 | 108 |
| Muddy mangrove pneumatophores and trunk | 351 | 499 |
| Muddy mangrove pneumatophores | 106 | 106 |
| Bare substratum | 131 | 155 |
| Mangrove - mixed pneumatophore prop root | 152 | 183 |
| Rocky mangrove - large boulder and trunk | 79 | 79 |
| Rock shelf | 139 | 235 |
| Large boulder and pneumatophores | 96 | 117 |
| **Total** | **4505** | **15463** |


**Fig. 1.** DeepFish image samples across 20 different habitats.

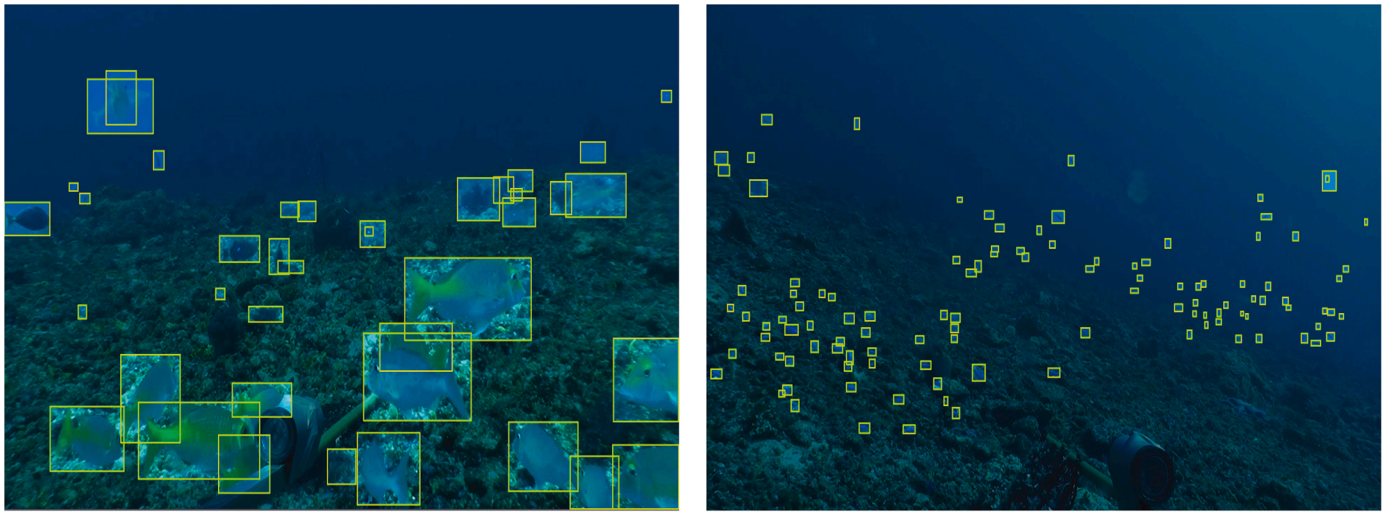



**Fig. 2.** OzFish image samples: (a) multitude of different sizes of fish, (b) multitude of tiny sizes of fish.

### 2.2. Basic YOLO models

You Only Look Once (YOLO) was first proposed (Redmon et al., 2016) as a deep learning model for object detection with high computation speed and, in a true sense, for real-time detection. It is a region-based convolutional neural network that combines region proposal network branch and classification stage into a single network. The YOLO model can directly predict the bounding boxes and their corresponding class probabilities with a single feed-forward network where the previous detection models take more time for region proposals (Girshick et al., 2014; Girshick, 2015; Tabassum et al., 2020). Subsequently, YOLOv2 (Redmon and Farhadi, 2017) was introduced with the idea of anchors from Faster-RCNN (Ren et al., 2015). After that, Joseph and Ali (2018) proposed YOLOv3 for object detection built on YOLO and YOLOv2. YOLOv3 consists of a backbone network called Darknet-53, an upsampling network, and three detection heads similar to the idea of Feature-Pyramid Network (FPN) (Lin et al., 2017) shown in Fig. 3. Darknet-53 adopts a residual block consisting of a $3 \times 3$ and $1 \times 1$ convolutional layer pair with a shortcut connection. The shortcut connections concatenate the intermediate layers of Darknet-53 to the layer right after the upsampling layer. Like FPN, YOLOv3 predicts objects in three different scales where FPN processes the image at a different spatial compression. A further improved version YOLOv4 (Bochkovskiy et al., 2020) was proposed which had three networks: YOLOv3 as the head, spatial pyramid pooling (He et al., 2015), path aggregation network (Liu et al., 2018) as the neck and CSPNet (Wang et al., 2020) as the backbone. A latest version YOLOv5 is also available via free repository.[1]

### 2.3. YOLO-Fish model

An overview of our proposed fish detection model YOLO-Fish is shown in Fig. 4. The design of the YOLO-Fish-1 model optimized the upsample step size. According to Gai et al. (2021), in YOLOv3, using upsampling is helpful to extract features and strengthen the feature fusion. During model training for the small object of low resolution like $16 \times 16$ is difficult to process when the input image is resized with $608 \times 608$. Upsample layer helps here by upsampling two times to twice the resolution, and then only it can hold the feature information of small objects. In our custom dataset, many images consist of tiny objects where after resizing to $608 \times 608$, the small objects can compress to $8 \times 8$ or $4 \times 4$. Thus, we changed the upsample step size 2 to 4 to upsampling the feature map four times. To establish small objects target detection layers at scale 3, we added upsampling output feature with the second residual block of Darknet-53 to get more small target feature information. In Fig. 5. the modified YOLO-Fish-1 model turns $38 \times 38 \times 128$ output feature map into $152 \times 152 \times 128$ to enhance feature fusion. The feature map of $152 \times 152 \times 128$ also is taken from the earlier second residual block of Darknet-53 and merged with the upsampling feature by concatenation.

Furthermore, the idea of Spatial Pyramid Pooling (SPP) (He et al., 2015) module was added next to Darknet-53 of YOLO-Fish-2, keeping the previous upsampling approach connected, as shown in Fig. 4. SPP net uses three scales for max-pooling operation, keeping input feature map and out feature map size the same. Upsampling was used to keep feature information of tiny objects where it can be lost in many cases. Nevertheless, since we are working on many habitats, including foggy, blurry environments. Images from those environments lead to a model for missed or inaccurate detection. Therefore, the SPP module shown in Fig. 6 can solve the problem here. According to Huang et al. (2020), it is a feature enhancement module, which extracts the main information of the feature map and performs stitching.

[1] https://github.com/ultralytics/yolov5.

## 3. Results and discussion

In this section, we firstly present the experimental setup, followed by evaluation strategies, results, performance analysis and discussion.

### 3.1. Experimental setup and evaluation

We trained and tested our Model in Azure (Cloud Computing Service) with the following specifications: NVIDIA Tesla K80 GPU, 64 bit 33 MHz, 6 cores, 56 GB RAM, CUDA v11.4, cuDNN v8.0.5, OpenCV v3.2.0. The detailed flowchart of dataset building, training and detection process of the YOLO-Fish model provides in Fig. 7.

Before training the models, we generated the anchor boxes for our custom dataset. For three scales of the YOLO detection layer, we got 9 clusters for $608 \times 608$ resolutions using $k$-means clustering (Hartigan and Wong, 1979) algorithm. Anchors were taken with the average IoU of 67.01% for OzFish, 71.78% for DeepFish, and 67.71% for the merged dataset. The learning rate is chosen to be 0.001 between 0 and 6000 iterations. The adjustment of the learning rate reduces training loss. The momentum and decay were 0.9 and 0.0005, respectively. For efficient memory use, we were set batch to 64 and subdivision 32. Moreover, a random flag was set to 1 to adjust model learning for different resolutions while training the model. We set max to 200 where the model can process position information of up to 200 objects in a single image. At last, we set the IoU ignore threshold to 0.5 to consider a valid detection where the proportion of the area of overlap between predicted bounding box $b_p$ and ground true bounding $b_t$ to the overall area exceeds 0.5.The corresponding formula Eq. (1) will be;

$$IoU = \frac{Area(b_p \cap b_t)}{Area(b_p \cup b_t)} \tag{1}$$

We used Precision, Recall, F1-score, and AP as evaluation parameters to evaluate our models. The formula of these parameters are shown in Eqs. (2)–(5).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{3}$$

$$F_1 = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{4}$$

Here, TP, FP, and FN abbreviation for True Positive, False Positive, and False Negative. True positive means correct prediction. A False Positive indicates incorrect detections that the box's level is Fish, but it was not Fish. On the other hand, False Negative indicates missed detection that the box does not contain a Fish where there was a Fish. F1-score combines precision and recall into a single measure. Average precision or AP measure overall performance of the models under different confidence thresholds, expressed in below;

$$AP = \sum (r_{n+1} - r_n) \underbrace{\max}_{\tilde{r} \geqslant r_{n+1}} p(\tilde{r}) \tag{5}$$

where $p(\tilde{r})$ is the measured Precision at Recall $\tilde{r}$.

### 3.2. Model performance on different datasets

The models were trained separately in each different dataset to see if the models could achieve better detection results than the state-of-the-art model. The models were tested using the image resolution of $608 \times 608$ pixels and set the batch size to 1 to maintain consistency with the training image resolution. The Precision, Recall, F1 -score and AP of the detected fish objects was calculated and compared with the YOLOv3 model. The experimental results are shown in Table 2 for OzFish, Table 3

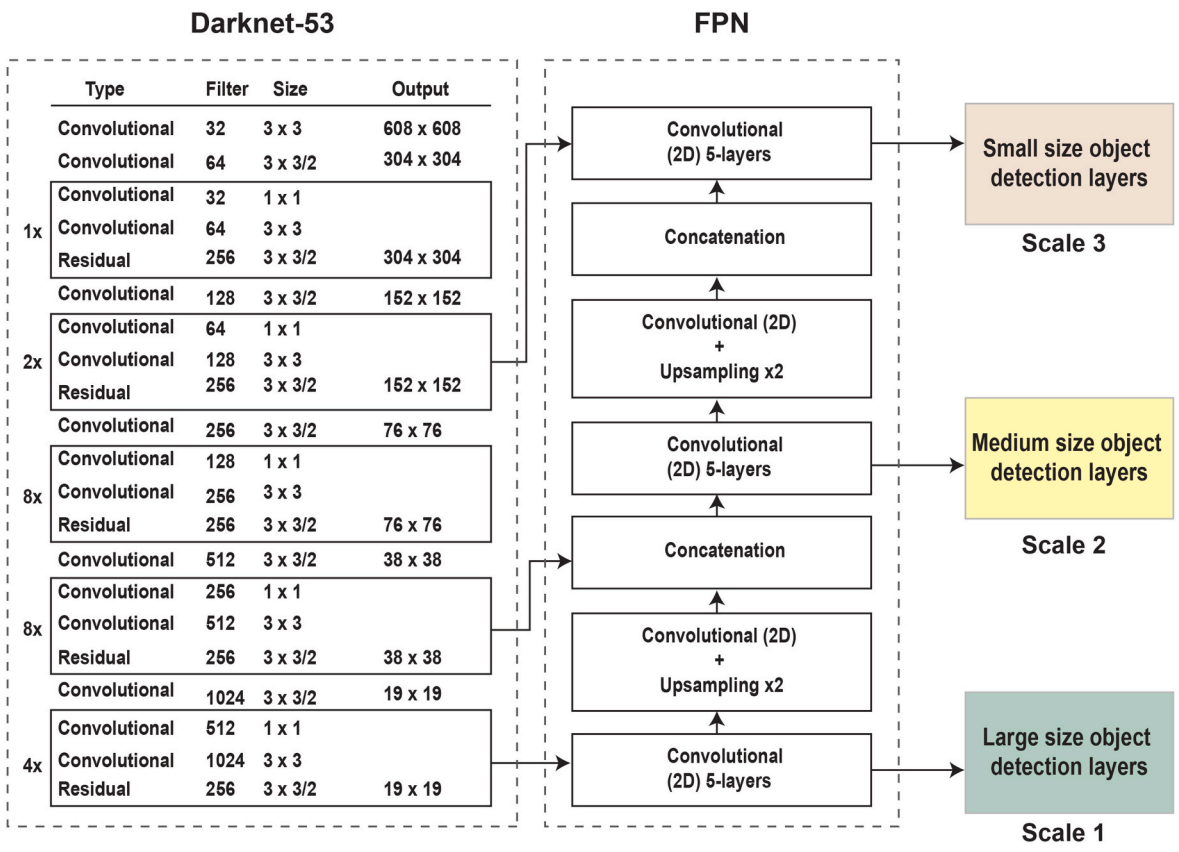



**Fig. 3.** YOLOv3 architecture.

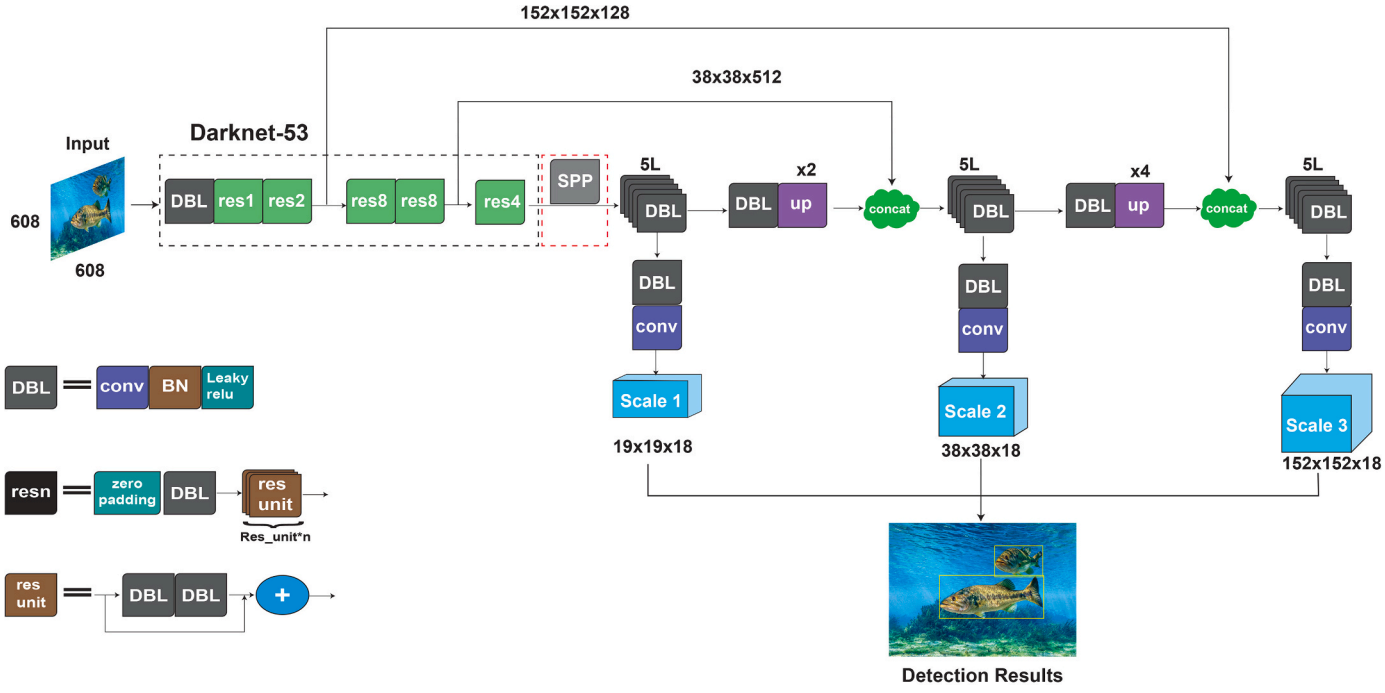



**Fig. 4.** Overview of YOLO-Fish model.

for DeepFish and Table 4 for merged dataset. In each table bold face font indicates best performance achieved by a model on the dataset.

From the results reported in Tables 2–4, we found that the performance of the models on DeepFish dataset is very high compared to other datasets. Moreover, the performance of the models on OzFish is comparatively low. The reason behind this is that Ozfish contained many tiny fish objects like a black dot where those are responsible for reducing the overall performance of the models. However, we merged

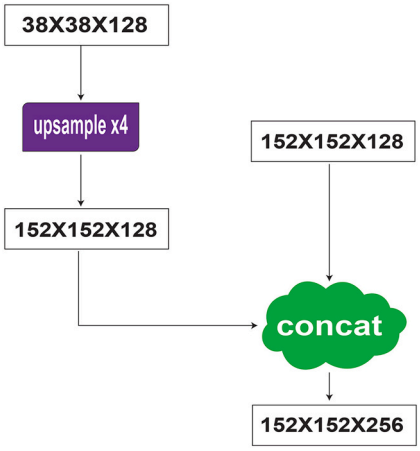



**Fig. 5.** The output feature map of upsampling.

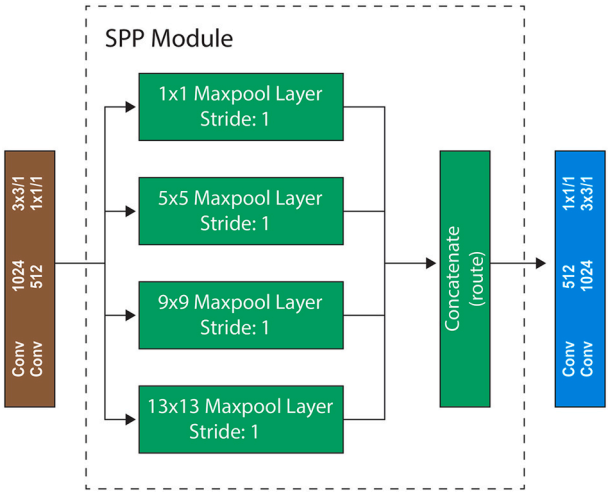



**Fig. 6.** Spatial pyramid pooling module.

two datasets to make models more robust by learning all necessary complexity from a realistic environment. Nevertheless, the YOLO-Fish models detect the number of fishes in the test dataset achieving good detection results. There are variations in the evaluated performance between the methods on different datasets. Compared with the results of the YOLOv3 model, the AP of YOLO-Fish-1 increased by 0.77% in Table 2, 0.14% in Table 3 and 1.29% in Table 4. Without a doubt, it shows that these models can detect small fish objects better due to changing upsample step size. YOLO-Fish-2 in Table 3 showed it is decreased by 0.27% compared to YOLOv3, although it is increased by 1.21% in Table 2 and 0.43% in Table 4. This is due to feature enhancement to the models provided by the SPP module to detect fish better. The F1-score remains the same for YOLOv3 and YOLO-Fish-1 models in Tables 2–4. Compared to YOLOv3, F1-score is increased by 1% in Table 2 for YOLO-Fish-2 but decreased again by 1% in Table 4. The precision increased every time for YOLO-Fish-1 and YOLO-Fish-2 in Tables 2–4. In the end, measuring all evaluation metrics specifically in case of AP, YOLO-Fish-1 shows the best detection performance among all the models.

We also trained the YOLOv4 model on our merged dataset to compare with YOLO-Fish. YOLOv4 achieved higher AP than other models as reported in Table 4. However, YOLOv4 failed to achieve good precision compared to other models and we see that it decreased 11% when we compare with YOLO-Fish-2. On the other hand, F1-score of YOLOv4 is very similar to all other models.

### 3.3. Complexity analysis between models

In addition to evaluating the performance of models based on accuracy and AP, we analyzed the models based on efficiency and size of parameters. It is necessary to develop an object detection that should be fast and light-weight along with improvements in terms of performance metrics. If the object detection model is not fast and light-weight it will fail to meet the objectives of real-time object detection and will also be difficult to train, test and deploy due to hardware resource constraints.

We evaluated our model's complexity with detection time, the total number of parameters, and billion floating-point operations per second (BFLOPs). We presented all the experimental values of each model for these parameters in Table 5. From Table 5, we note that YOLOv4 has worse parameters in each case than any other model. In the case of detection time, YOLO-Fish models take a higher time than YOLOv3 due to the model modification but still less than YOLOv4 (Bochkovskiy et al., 2020). On the other hand, the total number of parameters increased by around 2.5 million when the default YOLO model switched from version 3 to version 4. However, the parameters are increased by around 1 million for YOLO-Fish-2, which is still 1.5 million less than YOLOv4 where the parameters of YOLO-Fish-1 did not even increase despite having modifications. Having a huge number of parameters makes the model more complex and requires more time and memory to train in the fixed hardware setup. In our environmental setup, YOLOv3 took around 64–67 h to train whereas YOLOv4 took above 84 h. Moreover, both YOLO-Fish models strike the best in the case of BFLOPs from the experimental results. The experimental results of YOLO-Fish models show these models can process more than 46 billion floating-point operations than YOLOv4 per second. It clearly shows that YOLO-Fish models can process more BFLOPs than YOLOv3 and YOLOv4 in the same hardware unit.

### 3.4. Comparative analysis of models in different scenarios

We detected two separate unknown images from two datasets judging the capabilities of the models trained on the merged dataset. By looking at these detected images, we get a real idea about the performance of our models. The first image is taken from one habitat of DeepFish, where fish objects are hardly visible even to a human. The improvement in the model performance can be seen as the missed fish detections in YOLOv3 (Fig. 8(a)) is found before modification in YOLO-Fish-1 (Fig. 8 (b)) and YOLO-Fish-2 (Fig. 8(c)) marked by a circle. Also, in Fig. 8(c), YOLO-Fish-2 increased the percentage detection compared to YOLO-Fish-1 (Fig. 8(b)). On another image of OzFish, YOLO-Fish-1 outperformed the YOLOv3 to detect tiny objects in Fig. 9(b), whereas YOLOv3 (Fig. 9(a)) missed many of the tiny fish objects. Moreover, YOLO-Fish-1 (Fig. 9(b)) and YOLO-Fish-2 (Fig. 9(c)) avoided inaccurate detections flawlessly, whereas YOLOv3 has inaccurate detections in Fig. 9(a) marked by circle, although YOLO-Fish-2 performed a little worse than YOLO-Fish-1 in this particular environment to detect tiny fish. Overall visualizations show that YOLO-Fish-1 is better for detecting such tiny objects that were impossible to detect for the state-of-the-art model, and YOLO-Fish-2 works better than YOLOv3 to differentiate the sea background to detect a fish object.

### 3.5. Discussion

Real-time fish detection has been considered in different types of research in the last couple of years to monitor fish abundance in different habitats, for marine ecological research, and to maintain sustainable fisheries. The aim is to get an optimal and robust underwater fish detection model. While many of the works target a single or specific environments and single fish species, others target multiple environments and various species of fish. We have seen several deep learning-based algorithms or computer vision based methods have emerged among available successful techniques (Salman et al., 2020; Jalal et al.,

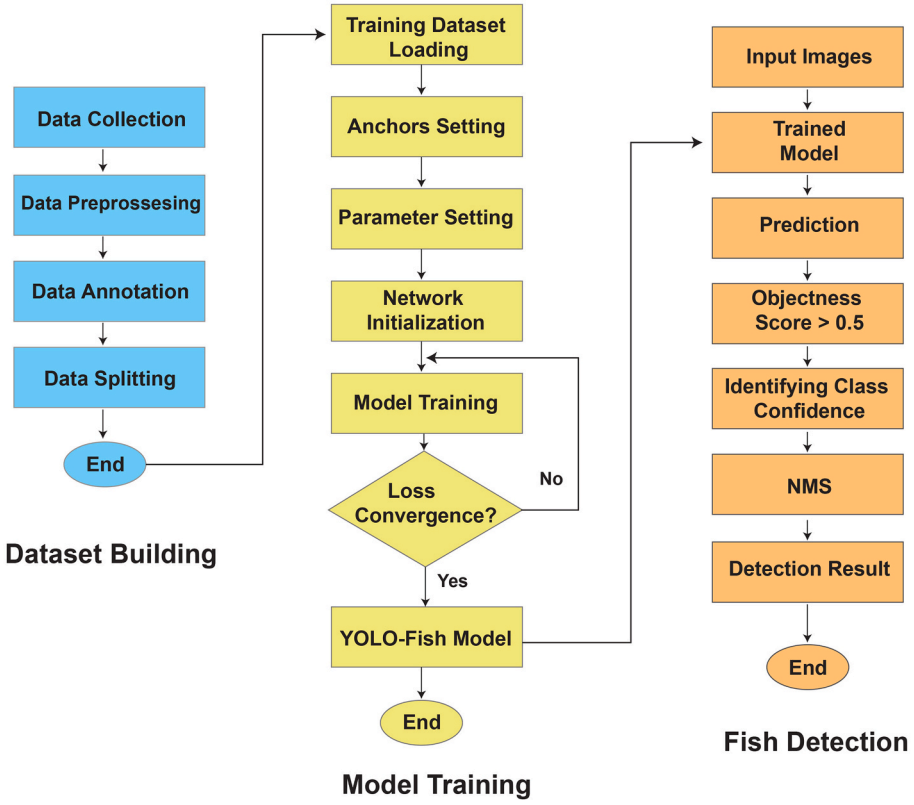



**Fig. 7.** YOLO-Fish model flowchart for dataset, training and detection process.

**Table 2**
Different model performance evaluation on OzFish dataset.

| Model | Precision (%) | Recall (%) | F1-score (%) | AP (%) |
|---|---|---|---|---|
| YOLOv3 | 80 | **65** | 72 | 69.13 |
| YOLO-Fish-1 | 82 | 64 | 72 | 69.9 |
| YOLO-Fish-2 | **83** | 64 | **73** | **70.34** |

**Table 3**
Different model performance evaluation on DeepFish dataset.

| Model | Precision (%) | Recall (%) | F1-score (%) | AP (%) |
|---|---|---|---|---|
| YOLOv3 | 94 | **94** | **94** | 96.01 |
| YOLO-Fish-1 | **95** | 93 | **94** | **96.15** |
| YOLO-Fish-2 | **95** | 93 | **94** | 95.74 |

**Table 4**
Different model performance evaluation on merged dataset.

| Model | Precision (%) | Recall (%) | F1-score (%) | AP (%) |
|---|---|---|---|---|
| YOLOv3 | 87 | 67 | 76 | 75.27 |
| YOLO-Fish-1 | 89 | 66 | 76 | 76.56 |
| YOLO-Fish-2 | **91** | 64 | 75 | 75.7 |
| YOLOv4 | 80 | **76** | **78** | **81.02** |

2020; Veiga et al., 2022).

The main contribution of this paper is not only the improvements of the underwater fish detection over the state-of-the-art models but also to introduce a large-scale dataset consisting of large-variant image samples from multi-type of habitats. We merged two different datasets from classification domain where each dataset carries different types of specialty in different cases that are already discussed in detail in Section 2.1. In our experiments, we first worked on the Deepfish dataset and

**Table 5**
Complexity comparison of different models.

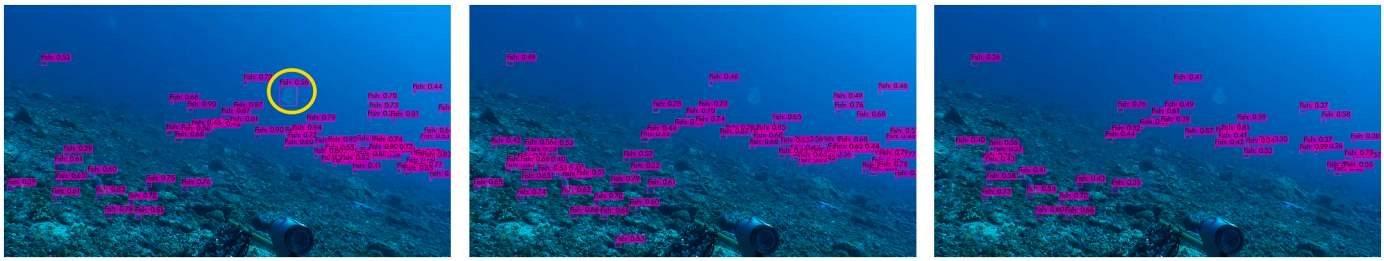
| Habitat | Detection Time (ms) | Total Number of Parameters | BFLOPs |
|---|---|---|---|
| YOLOv3 | 43.29 | 61, 576, 342 | 139.496 |
| YOLO-Fish-1 | 49.91 | 61, 559, 958 | 173.535 |
| YOLO-Fish-2 | 51.06 | 62, 610, 582 | 174.343 |
| YOLOv4 | 53.21 | 64, 003, 990 | 127.232 |

developed the fish detection model that can handle the complexity of multiple fish types in nearly all marine habitats. Moreover, we achieved better results in terms of Average Precision(AP) over the state-of-the-art model in Table 3. However, when we examined the detection results of those models in unknown environments, we discovered that they cannot detect small or tiny fish. Therefore, to address this issue with robustness, we bring out another large-scale dataset called OzFish with huge variants of fish types and sizes between tiny, medium, and large. Finally, merging the two datasets helped us transform a new benchmark dataset and expand the advancement scope into developing a robust fish detection model.

In the literature review, we showed several research on fish detection mainly focused on improving the methodological approach without considering the limitation of datasets. An excellent contribution was found in Salman et al. (2019) to improve the classification models for fish detection on the most popular fish dataset, Fish4Knowledge. They proposed a method to improve fish segmentation in an unconstrained underwater environment using pixel-wise posteriors on adaptive background subtraction from the Gaussian mixture model, which performs better than existing solutions achieving an F1-score of 84.28%. However, the dataset used in this research is Fish4Knowledge, whose limitations are already highlighted in the literature review. The authors in a consecutive research (Jalal et al., 2020), heavily focused on the dataset

(a) (b) (c)

**Fig. 8.** Comparison of YOLO-Fish models for (a) YOLOv3 (b) YOLO-Fish-1 and (c) YOLO-Fish-2 detection results on a DeepFish image.



(a) (b) (c)

**Fig. 9.** Comparison of YOLO-Fish models for (a) YOLOv3 (b) YOLO-Fish-1 and (c) YOLO-Fish-2 detection results on a OzFish image.

with the same goal: making a robust fish detection model. Their proposed hybrid solution to combine optical flow and Gaussian mixture models with YOLO achieved improved results. However, the two datasets used in this research also have limitations compared to our dataset, although the datasets are extensive. One of the two datasets is LCF-15, extracted from Fish4Knowledge, and the second one is their own UWA dataset, consisting of many image samples with poor fish visibility. However, there are no image samples with labels provided from the UWA dataset where we could see every fish instance, including small to large fish that are annotated. Moreover, in the detection results, we saw that their proposed model failed to detect many small fish instances. The detection F1-score of 91.2% was achieved on the UWA dataset, which is good but can not be considered a robust fish detector compared with our YOLO-Fish model. In our experiments, we showed how large number of tiny fish instances are responsible for reducing the overall performance of a model. The OzFish dataset we used was part of the research from Veiga et al. (2022), where they performed automated labeling on the dataset during the training process to employ data for fish detection. The proposed automated labeling method filters and minimizes the noise data in datasets such as negligible small bounding boxes in OzFish. However, Bounding boxes of fish were generated by random sampling frames from the OzFish (Australian Institute Of Marine Science, 2020) video archive that had associated measurements and analyzed on the Amazon Sagemaker Ground Truth platform. The bounding boxes are combined results from multiple observers. In their results, the AP of 68.46% was achieved by adopting YOLOv4 trained on the OzFish dataset, which is still worse compared to the YOLO-Fish model, where the YOLO-Fish-2 model achieved an AP of 70.34% trained on OzFish.

The results of our research appeared to be competitive compared to Labao and Naval (2019). In Labao and Naval (2019), the authors used a dataset having more than 10000 fish objects in training set including many small fish. Among the three implemented deep neural network based object detection models proposed in their research, one of the models achieved the highest detection F1-score of 67.76% tested on unseen fish objects in videos that were not used for training. In comparison, the YOLO-Fish model achieved an F1 score of 76% which is significantly better considering dealing with a massive number of fish instances including numerous fish types from multiple and complex underwater backgrounds.

## 4. Conclusion

This research work introduces a new real-world environmental dataset as a benchmark suite consisting of two large-scale datasets, DeepFish and OzFish. We also present robust fish detection models modifying YOLOv3 trained on the merged dataset to solve the limitations of YOLOv3, a state-of-the-art model. The YOLOv3 model was unable to process the feature information of tiny objects in the dataset during training and could not distinguish fish from complex sea backgrounds in test detection. The modified YOLO-Fish models have been introduced to solve these problems and make detectors as intelligent as humans by reducing missed and inaccuracies. The experimental results show that the proposed YOLO-Fish models performed better than the state-of-the-art methods based on Average Precision. Since the datasets were collected from the marine environments, the model is applicable to monitoring marine fisheries. Fisheries resources in global are mainly divided into inland, coastal and marine cultures. Though the later versions of YOLO achieves similar performances, they suffer due to huge model complexity. It might be interesting to see how the recent developments in computer vision like attention based models (Hu et al., 2018) and transformers (Dosovitskiy et al., 2020) apply to this field. Also, the fish datasets could be enhanced by including more fish images from other inhabitants or cultures.

## Availability of the materials

Code and supplementary data to this article can be found at https://github.com/tamim662/YOLO-Fish.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Link to data and code is provided in the repository mentioned in the manuscript

## References


Adiwinata, Y., Sasaoka, A., Agung Bayupati, I.P., Sudana, O., 2020. Fish species recognition with faster r-cnn inception-v2 using qut fish dataset. Lontar Komputer: Jurnal Ilmiah Teknolologi Informasi 11 (3), 144.

Anantharajah, Kaneswaran, Ge, ZongYuan, McCool, Chris, Denman, Simon, Fookes, Clinton, Corke, Peter, Tjondronegoro, Dian, Sridharan, Sridha, 2014. Local inter-session variability modelling for object classification. In: IEEE Winter Conference on Applications of Computer Vision. IEEE, pp. 309–316.

Australian Institute Of Marine Science, 2020. Ozfish dataset - machine learning dataset for baited remote underwater video stations.

Bochkovskiy, Alexey, Wang, Chien-Yao, Mark Liao, Hong-Yuan, 2020. Yolov4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

Cai, Kewei, Miao, Xinying, Wang, Wei, Pang, Hongshuai, Liu, Ying, Song, Jinyan, 2020. A modified yolov3 model for fish detection based on mobilenetv1 as backbone. Aquacult. Eng. 91, 102117.

Cutter, George, Stierhoff, Kevin, Zeng, Jiaming, 2015. Automated detection of rockfish in unconstrained underwater videos using haar cascades and a new image dataset: labeled fishes in the wild. In: 2015 IEEE Winter Applications and Computer Vision Workshops. IEEE, pp. 57–62.

Dosovitskiy, Alexey, Beyer, Lucas, Kolesnikov, Alexander, Weissenborn, Dirk, Zhai, Xiaohua, Unterthiner, Thomas, Dehghani, Mostafa, Minderer, Matthias, Heigold, Georg, Gelly, Sylvain, et al., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Fao, 2020. The state of world fisheries and aquaculture 2020. Sustainability in action. Rome.

Fisher, R., Boom, B., Huang, P. Preliminary experiments with the fish4knowledge dataset. Algae, 49165 (49370), 99–58.

Gai, Wendong, Liu, Yakun, Zhang, Jing, Jing, Gang, 2021. An improved tiny yolov3 for real-time object detection. Syst. Sci. Control Eng. 9 (1), 314–321.

Girshick, Ross, 2015. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 1440–1448.

Girshick, Ross, Donahue, Jeff, Darrell, Trevor, Malik, Jitendra, 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587.

Hartigan, John A., Wong, Manchek A., 1979. Algorithm as 136: a k-means clustering algorithm. J. R. Stat. Soc. Ser. C (Appl. Stat.) 28 (1), 100–108.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, Sun, Jian, 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. 37 (9), 1904–1916.

Horwath, James P., Zakharov, Dmitri N, Mégret, Rémi, Stach, Eric A., 2020. Understanding important features of deep learning models for segmentation of high-resolution transmission electron microscopy images. npj Comput. Mater. 6 (1), 1–9.

Huang, Yi-Qi, Zheng, Jia-Chun, Sun, Shi-Dan, Yang, Cheng-Fu, Liu, Jing, 2020. Optimized yolov3 algorithm and its application in traffic flow detections. Appl. Sci. 10 (9), 3079.

Hu, Jie, Shen, Li, Sun, Gang, 2018. Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132–7141.

Jalal, A., Salman, A., Mian, A., Shortis, M., Shafait, F., 2020. Fish detection and species classification in underwater environments using deep learning with temporal information. Ecol. Inform. 57, 101088.

Redmon, Joseph, Farhadi, Ali, 2018. Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767.

Labao, A., Naval Jr, P., 2019. Cascaded deep network systems with linked ensemble components for underwater fish detection in the wild. Ecol. Inform. 52, 103–121.

Li, Kunyi, Cao, Lu., 2020. A review of object detection techniques. In: 2020 5th International Conference on Electromechanical Control Technology and Transportation (ICECTT). IEEE, pp. 385–390.

Li, Xiu, Shang, Min, Qin, Hongwei, Chen, Liansheng, 2015. Fast accurate fish detection and recognition of underwater images with fast r-cnn. In: OCEANS 2015-MTS/IEEE. IEEE, Washington, pp. 1–5.

Li, Xiu, Shang, Min, Hao, Jing, Yang, Zhixiong, 2016. Accelerating fish detection and recognition by sharing cnns with objectness learning. In: OCEANS 2016-Shanghai. IEEE, pp. 1–5.

Li, Xiu, Tang, Youhua, Gao, Tingwei, 2017. Deep but lightweight neural networks for fish detection. In: OCEANS 2017-Aberdeen. IEEE, pp. 1–5.

Lin, Tsung-Yi, Dollár, Piotr, Girshick, Ross, He, Kaiming, Hariharan, Bharath, Belongie, Serge, 2017. Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117–2125.

Liu, Shu, Qi, Lu, Qin, Haifang, Shi, Jianping, Jia, Jiaya, 2018. Path aggregation network for instance segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8759–8768.

Nour Eldeen, M., Khalifa, Mohamed Hamed, Taha, N., Hassanien, Aboul Ella, 2018. Aquarium family fish species identification system using deep neural networks. In: International Conference on Advanced Intelligent Systems and Informatics. Springer, pp. 347–356.

Redmon, Joseph, Farhadi, Ali, 2017. Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263–7271.

Redmon, Joseph, Divvala, Santosh, Girshick, Ross, Farhadi, Ali, 2016. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788.

Ren, Shaoqing, He, Kaiming, Girshick, Ross, Sun, Jian, 2015. Faster r-cnn: towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 28, 91–99.

Sabottke, Carl F., Spieler, Bradley M., 2020. The effect of image resolution on deep learning in radiography. Radiol.: Artif. Intell. 2 (1), e190015.

Saleh, Alzayat, Laradji, Issam H., Konovalov, Dmitry A., Bradley, Michael, Vazquez, David, Sheaves, Marcus, 2020. A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis. Sci. Rep. 10 (1), 1–10.

Salman, A., Maqbool, S., Khan, A., Jalal, A., Shafait, F., 2019. Real-time fish detection in complex backgrounds using probabilistic background modelling. Ecol. Inform. 51, 44–51.

Salman, Ahmad, Siddiqui, Shoaib Ahmad, Shafait, Faisal, Mian, Ajmal, Shortis, Mark R., Khurshid, Khawar, Ulges, Adrian, Schwanecke, Ulrich, 2020. Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system. ICES J. Mar. Sci. 77 (4), 1295–1307.

Szegedy, Christian, Toshev, Alexander, Erhan, Dumitru, 2013. Deep neural networks for object detection.

Tabassum, Shaira, Ullah, Md Sabbir, Al-Nur, Nakib Hossain, Shatabda, Swakkhar, 2020. Native vehicles classification on Bangladeshi roads using CNN with transfer learning. In: Proceedings of the 2020 IEEE Region 10 Symposium (TENSYMP), pp. 40–43.

Veiga, R., Ochoa, I., Belackova, A., Bentes, L., Silva, J., Semião, J., Rodrigues, J., 2022. Autonomous Temporal Pseudo-Labeling for Fish Detection. Appl. Sci. 12, 5910.

Wang, Chien-Yao, Mark Liao, Hong-Yuan, Wu, Yueh-Hua, Chen, Ping-Yang, Hsieh, Jun-Wei, Yeh, I-Hau, 2020. Cspnet: a new backbone that can enhance learning capability of cnn. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp. 390–391.

Wang, Wenkai, He, Bingwei, Zhang, Liwei, 2021. High-accuracy real-time fish detection based on self-build dataset and rird-yolov3. Complexity 2021.

Zhao, Zhong-Qiu, Zheng, Peng, Shou-tao, Xu., Xindong, Wu., 2019. Object detection with deep learning: a review. IEEE Trans. Neural Netw. Learn. Syst. 30 (11), 3212–3232.