Journal of Integrated
SCIENCE & TECHNOLOGY

A deep learning approach for underwater fish detection

Vrushali Pagire,[1,*] Anuradha C. Phadke,[1] J Hemant[2]

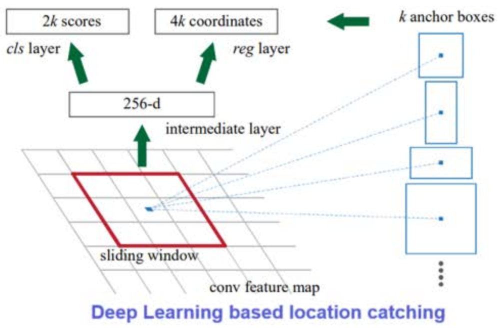
[1]Electronics and Communication Engineering, Dr. Vishwanath Karad MIT World Peace University, Pune. India. [2]Karunya Institute of Technology and Sciences, Karunya Nagar, Coimbatore, Tamil Nadu. India.

Received on: 28-Aug-2023, Accepted and Published on: 07-Dec-2023

Article

ABSTRACT

Identifying moving objects in video sequences is crucial for various applications, including underwater surveillance, biomedical detection, threat identification, defence, and navy. When comparing images and videos captured in an oceanic environment to those captured in an air, the physical properties of the water medium usually cause degradation, leading to unstable or lost features. Although there are many applications of underwater object detection, researchers find it difficult to extract objects from underwater images due to problems like water body turbidity, blurring, and low image quality. In static background conditions, the background remains stationary, while in dynamic background conditions, both the background and foreground exhibit motion, making it difficult to differentiate between them. The differentiation of both the background and the foreground object is very difficult in dynamic as compared to static. Therefore, the suggested system offers a deep learning-based solution for underwater fish detection that employs three models: YOLOv3, SSD Mobile net v2, and Faster R-CNN ResNet50, all of which were trained on a bespoke dataset called Fish4knowledge. The algorithms have been taught to recognize and reliably pinpoint fish species in underwater photos and videos. To improve performance, data pre-treatment, model selection, and hyperparameter adjustment are carried out. The best-performing model is chosen after evaluation on a different validation dataset. Model updates and adaption to changing undersea conditions are required for long-term accuracy and performance enhancement. The precision value for the Faster R-CNN ResNet50, YOLOv3 and SSD MobileNetV2 is 45.06%, 79% and 98.21% respectively. The research results demonstrate that the SSD MobileNetV2 model gives the highest precision value as compared to the YOLOv3 and Faster R-CNN ResNet50 models.



Keywords: Deep learning, anchor box, object detection, underwater images, YOLOv3, deep learning, Faster R-CNN, SSD, MobileNetV2

INTRODUCTION

The identification of moving items in video arrangements is among the main challenges for many video processing systems. Numerous uses, including underwater surveillance, biomedicine, the identification of dangers, and the detection of unauthorized entrants in the defence and navy, all benefit greatly from the automation of detecting systems.[1] Research and surveillance applications can take new directions through the analysis of underwater images. However, the health of the ocean, water quality, biodiversity, climatic change, and the impact of the food chain, ocean pollution, invasive species, sustainability, and biodiversity of the species that survive underwater may all be ascertained through the analysis of underwater video. When compared to images captured in the air, images and videos captured in a marine normally suffer from the physical characteristics of the water medium. Since light is strongly attenuated when it travels through water, as depth increases, captured scenes become lower contrasty and obscure. As depth increases, not only does brightness decrease, but colour quality also begins to deteriorate gradually based on colour wavelength. Images and videos appear bluish because blue light has a shorter wavelength and travels longest in water. There are many applications of underwater object detection, however extracting items from underwater images seems to be a difficult task for researchers due to problems like water body

*Corresponding Author: Vrushali Pagire
Email: pagirevrushali27@gmail.com

Cite as: *J. Integr. Sci. Technol., 2024, 12(3), 765. URN:NBN:sciencein.jist.2024.v12.765*

turbidity, blurring, and low image quality. In the proposed work fish is a moving object. Fish are a crucial component of human culture, industry, and marine ecosystems. Over three billion individuals around the world eat fish regularly.[1] Pollution, excessive fishing, and habitat loss, on the other hand, cause species to become extinct or be replaced. To assist in the preservation and regulatory actions that guarantee good stocks of fish and environments, it is crucial to monitor the abundance and frequency of fish species.[2,3] Based on the scope, the classification of fish species can be broadly classified into three application categories[4,5]:

• Identification of fish species on dead fish. (For instance, how industries classify conveyor belts.)

• Identification of fish species in artificial habitats, such as aquariums and water tanks.

• Identification of fish types in their native environments, such as oceans and seas.

For monitoring at-risk fish, sonar, and video data paired with the latest breakthroughs in machine learning offer a possible substitute for aggressive fish tags.[6] Several surveillance systems use hydroacoustic sensors to identify fish however because of inadequate resolution, they are restricted to contrasting relative biomass across period or recognizing big fish schools. Additionally, complicated sorting is required by relative biomass techniques to eliminate distortion from non-fish resources, such as tidal embedded air, that might else seem to be a huge number of fish.[7] An automated, comprehensive monitoring system might greatly cut labour costs while boosting throughput and accuracy, hence improving the precision of approximations of fish stocks, fish distribution, and biodiversity in common.[8] Effective computer vision (CV) processes are required for the implementation of such systems. As an outcome, a substantial study has been undertaken on the implementation of observing tools and approaches based on CV algorithms for evaluating how fish exploit diverse maritime settings and discriminating among fish species.[9] In the disciplines of image analysis and CV, deep neural networks (DNNs) have regularly generated cutting-edge outcomes in a wide range of uses, from medicine to agriculture.[10-15]

Particularly, the footage is made up of visuals or frames that have undergone image analysis processing. As a result, picture and video-based monitoring tasks can be performed using DL models like as convolutional neural networks (CNNs), which use an image (frame) as input. As a result, the approaches described for image-based jobs apply to both photos and videos.[16] Deep learning techniques are now often employed for fish visual feature detection and classification.[17] According to An, D. et al.[18] deep learning methods have established themselves as a significant knowledge for creating smart aquaculture systems and providing crucial data for changing and improving fish breeding. As of right now, there are two types of deep convolutional neural networks: (1) two-stage techniques, and (2) one-stage approaches.[19] The two-stage techniques are correct, but because of the complex computation required, they are too sluggish.[20] For identifying fish behaviour based on in-the-moment fish movements, one-stage techniques have proven to be quite effective. The YOLO (You Only Look Once) family's one-stage system combines the region proposal network with the detection of objects,[21] RetinaNet[22], or Single Shot Multi-Box Detector (SSD),[23] to streamline detection and speed up computations. For instance, Sung, et al.[24] monitored fish in an actual breeding farm using a YOLOv3 network with the bottleneck of MobileNetV1.

To devalue the training model, the conventional DenseNet model was initially swapped out for a MobileNet model. The image features were then extracted using a smaller dataset made up of 16 species that were obtained from "ImageNet". It may be seen by contrasting the backbone with other bone procedures that the suggested technique accurately estimates the number of fish present. Cao, et al.[25] proposed an online YOLO-based underwater video fish recognition system in another study. For fish detection, the network has an accuracy of 0.93, an intersection over the union of 0.634, and a frame rate of 16.7 FPS. Liu, et al.[26] implemented reliable, real-time recognition of live crabs submerged using a lightweight MobileNetV2 as the foundation of the SSD. The test results demonstrate that the faster MSSDLite is superior to conventional SSD in terms of performance. Among the foremost standard single-stage systems for tracking fish behaviour is the YOLO series since its compact system may be installed on a device at the edge (like a Raspberry Pi).[27] The performance of the classic YOLO series is excellent to forecast tiny goals at multiple scales, but because the characteristic map used for detection remains unchanged without implementing the field of reception into account, it performs poorly for large and medium-sized objects.[28]

Considering the foregoing, the overarching objective of the present task aims to create a completely automated system for tracking fish activity in an integrated hybridization environment as well as to offer solid conceptual justification for smart, online supervision in aquaculture, ensuring fish welfare and productivity. This is accomplished by describing and quantifying fish activity from visual images employing an inexpensive submarine camera technology. The network combines the enhanced YOLOv3-Lite with the MobileNetV2 backbone, which offers an improved spatial pyramid pooling block and a loss function based on the intersection over union (IoU), to address the shortcomings of classic YOLO families. The primary contributions of this paper are

- Moving object detection in video sequences is crucial for various applications, therefore, the suggested method offers a deep learning-based solution for underwater fish recognition that employs three models: YOLOv3, SSD MobileNetV2, and Faster R-CNN ResNet50.
- The YOLOv3, SSD MobileNetV2 and Faster R-CNN ResNet50 models are implemented for fish recognition in dynamic background and performance metric parameters are measured. These three models trained on Fish4knowledge dataset.
- The algorithms have been taught to recognize and reliably pinpoint fish species in underwater photos and videos.
- The analysis and comparison of the experimental results with other state of the art methods.

LITERATURE SURVEY

Knausgard, et.al.[29] provide a two-step, pre-filter-free deep learning method for identifying and classifying temperate fishes.

The YOLO object recognition process is used in the primary stage to find distinct fish in a frame. Each fish is classified in the second stage using a CNN with a squeeze-and-excitation (SE) design. Accuracy is increased by using transfer learning. On a publicly available dataset, the network is trained using ImageNet and the fish classifier, with post-training weights derived from pre-training. The approach demonstrates its viability with a larger dataset by achieving a modern accuracy of 99.27% utilizing the pre-training model and 83.68% and 87.74% with and without image augmentation. Furthermore, deep learning models may fail to adjust to modifications in the environment, such as illumination, water quality, or fish behaviour patterns, which might have an impact on their performance.

Jalal et al.[30] present a fusion technique that merges optical flow and Gaussian mixture models with a YOLO deep neural network to recognize as well as categorize fish in unconstrained underwater footage. Initially, YOLO-based object recognition systems were utilized to catch only static as well as visible fish instances. Using time-based data obtained from Gaussian mixture models and optical flow, overwhelmed YOLO's limitation and enabled it to recognize liberally swimming fish that are masked in the background. On both datasets, it attains fish recognition F-scores of 95.47% and 91.2%, respectively, and fish class categorization accuracies of 91.64% and 79.8%. However, there are several cases where the suggested method fails to recognize fish or misclassifies types owing to great differences in submerged settings.

Testolin et al.,[31] offer a deep-learning method for detecting a swimming fish pattern from the replications of an active sound emitter. This paper uses CNN, which allows for the concurrent labelling of a huge barrier of signal models, to enable real-time detection. This permits the assembly of the reflecting signal from the moving objective to be captured and separated from clutter reflections. When evaluated on actual signals, the network skilled on simulated forms demonstrated non-trivial finding abilities, implying that transfer learning could be a feasible strategy in these settings, where labelled information is frequently unavailable. However, by training the network directly on real-world reflections using data augmentation approaches, it can achieve a more favourable precision-recall trade-off, nearing an optimal detection bound. The proposed model does not represent an automated or efficient approach for monitoring marine ecosystems.'

Chhabra, et al.,[32] present a fish categorization system that operates in the normal submerged atmosphere and aids in the detection of fish with medical or nutritious value. This work employs a fused deep learning model that employs a pre-trained VGG16 model for feature extraction and a Stacking ensemble model to recognize and categorize fishes from photos. The system was tested using separate classes of 8 diverse fish species, such as cod and mackerel, and 435 photos. The author built their dataset because there was no public dataset linked to these fish species. When compared to various existing processes (kNN, SVM, RF, and Tree), the proposed approach beat them with a categorization accuracy of 93.8%.

However, the process became more difficult due to the lack of a previous dataset on fish with nutritional as well as beneficial relevance.

Han, et al.[33] investigate a deep learning-based jellyfish recognition technique that depends on CNN theory and digital image processing technologies. Using underwater image processing methods, it detects 10 jellyfish and fish species. The results suggest that dark channel prior, quadratic merging gray world, seamless reflection, and contrast-limited adaptive histogram equalization algorithms increase image quality. With an ordinary detection accuracy of 74.96%, the GoogLeNet backbone network exceeds AlexNet in jellyfish categorization. The study contributes to the development of jellyfish monitoring technology and marine biologist research by providing a hypothetical and methodological framework for real-time underwater jellyfish optical imaging. However, it is difficult to classify and detect jellyfish photos owing to the low superiority of underwater imagery and the diversity of jellyfish.

Methodology

This work aims to investigate the use of three sophisticated object detection models YOLOv3, SSD MobileNetV2, and Faster R-CNN ResNet50 for the detection of underwater fish using a specific dataset (Fish4knowledge). The distinctive qualities of each model are used to overcome the difficulties of fish detection in changing aquatic conditions. YOLOv3 is capable of catching fish that are moving quickly since it has real-time performance and quick detection. The SSD MobileNetV2 version exhibits effectiveness and multi-scale detection, allowing precise identification of fish of various sizes. Accuracy and accurate object localization are prioritized by the quicker R-CNN ResNet50, which helps with the detailed recognition of each fish. This study advances our knowledge of aquatic ecosystems and conservation efforts by shedding light on the best model selection for precise, real-time, and resource-efficient underwater fish detection. The overall system flow description of the suggested method is presented in Fig 1.

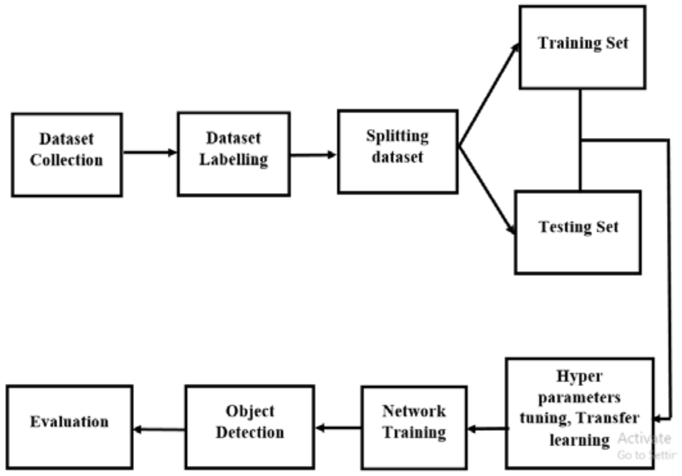


Figure 1: Flow Chart for Dataset Management and Detection.

3.1 System Specifications

The suggested deep learning-based technique is tested against the Fish4Knowledge dataset using simulations, data pre-processing, and Python 3 modules (such as the NumPy, pandas, seaborn, and Sk learn packages). The model was created using

Tensor Flow 2.10 and the Keras Python library. The proposed deep-learning approaches were used for underwater acoustic image analysis for the automatic detection of moving underwater fish. The dataset containing underwater films as well as ground truth frames was used to evaluate the system.

3.2 Dataset Description

The Fish4Knowledge audio-visual collection of data is provided in a video file version. The audio-visual files are transformed into images, which are subsequently labelled with the labelling tool. A total of 2500 frames were acquired and manually labelled using the LableImg tool. The frames have been labelled using the YOLO design that incorporates the left bottom as the foundation and includes the object class, bounding box coordinates, and the height and width of the image. The dataset was split among 30% testing and 70% training.

3.3 Deep Learning Method for Detecting Underwater Fish

In this following section, the three pre-defined deep learning models such as YOLOv3, SSD MobileNetV2, and Faster R-CNN are discussed in detail for underwater fish detection by using an improved dataset called Fish4knowledge.

3.4 YOLOv3 (You Only Look Once)

As demonstrated in Fig. 2, the YOLO architecture is built on CNN. Before YOLOv3, there were three previous versions. The single-phase finder model was first implemented as YOLOv1, which used batch normalization, a leaky ReLU activation operation, and reduction layers of size 1x1 in addition to convolutional layers of length 3x3. The system is made by using twenty-four convolutional layers for feature extraction and two fully connected (FC) layers for predicting bounding boxes as well as class probability. The result is a 7x7x30 tensor made up of bounding boxes. This system has been trained to identify 49 objects, though, it has a great degree of fault when it comes to localizing them.

YOLOv2 is an advanced form of YOLOv1, with an emphasis on condensed localization faults. YOLOv2 eliminated the final FC layers as well as implemented batch normalization to entire convolutional layers, resulting in network resolution independence as well as lesser localization fault. YOLOv2 used darknet-19, a network with 19 levels and an additional 11 layers, to find objects.

Because neither of the earlier YOLO methods could identify more than twenty classes, the YOLO9000 model was designed to identify and categorize more items and classes. These systems were later refined by using additional characteristics such as residual blocks, skip connections, and up-sampling as well as dubbed YOLOv3, which used a 53-layered network trained on the Fish4Knowledge dataset.

YOLOv3 is not intended to be an image classifier but a multiscale detector. As a result, a detection head is added to this design in place of the classification head for object detection. The result is a vector with the bounding box coordinates and probability classes from this point forward. The foundation of YOLOv3 is Darknet-53, a method for training 53-layer neural networks. A total of -106 layers of fully convolutional architecture—106 in all—are added on top of it for the object detection task. YOLOv3 employs 3 feature maps of various scales for target detection thanks to its multiscale feature fusion layers.

3.4.1. Bounding Box Prediction

Because YOLOv3 employs a distinct channel for characteristics abstraction, the entire frame is given to the convolutional system, which generates a square outcome named the network where the bounding boxes are fastened. The network cell and anchor have the same centroid. $t_x, t_y,\ t_w, t_h$, Objectness scores, and class likelihood are all predicted by the YOLO algorithm. The objectness score indicates the degree of certainty that an object exists within the enclosing box, while the class probability indicates whether the object belongs perfectly. The forecasts belong to the bounding box coordinates, with $(C_x,\ C_y)$ representing the upper-left angle in addition to P_h as well as P_w representing the breadth as well as height, as shown in Fig. 3 as well as evaluated as stated in Eq. 1-4.

$$b_x = \sigma(\ t_x) + C_x \tag{1}$$

$$b_y = \sigma\big(\ t_y\big) + C_y \tag{2}$$

$$b_w = P_w\, C^{t_w} \tag{3}$$

$$b_h \ = \ P_h \tag{4}$$

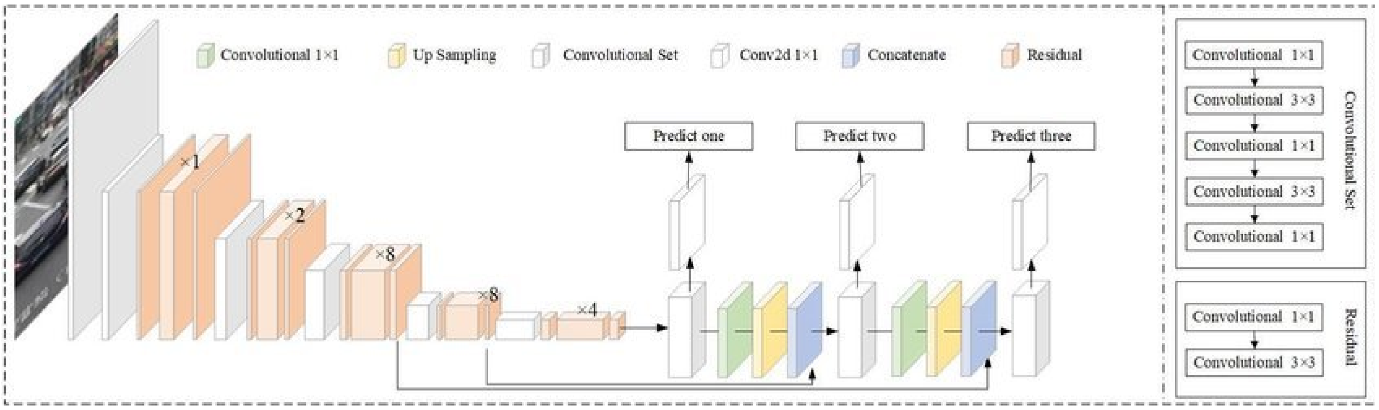


Figure 2: 106-layer fully convolutional architecture

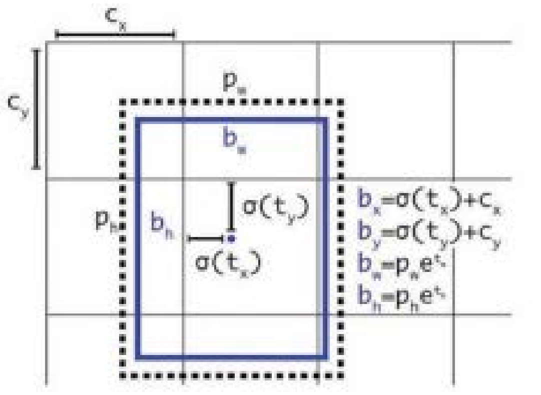


Figure 3: Bounding box

b_x denotes the x-axis, b_y denotes the y-axis, as well as $b_y$and b_h are the height and breadth. Logistic regression is used to compute the objectness score, which is a count of the overlying of ground truth and bounding box. Value "1" denotes a seamless overlap of the bounding box and ground truth or overlap over a threshold, while value "0" indicates that the overlap is a fault and is under a threshold, and the bounding box is disregarded.

Initially, the objectness score aids in the selection of the ideal bounding box. In general, bounding boxes having a higher objectness value than the threshold is eliminated initially and formerly examined for additional filtering. Most object detection algorithms have the issue of recognizing similar items at diverse times in multiple images, causing worse presentation. To tackle the problem of numerous detections of the same image, YOLO employs non-maximal suppression (NMS). NMS employs a specific function known as Intersection of Union (IOU) by specifying the lowest IOU threshold, which is typically set at 0.5. If $B1$ and $B2$ are dual bounding boxes, the IOU is calculated as the proportion of the joining of the ranges of $B1$ and $B2$ to the entire range of $B1$ and $B2$.

3.4.2 Prediction of Class

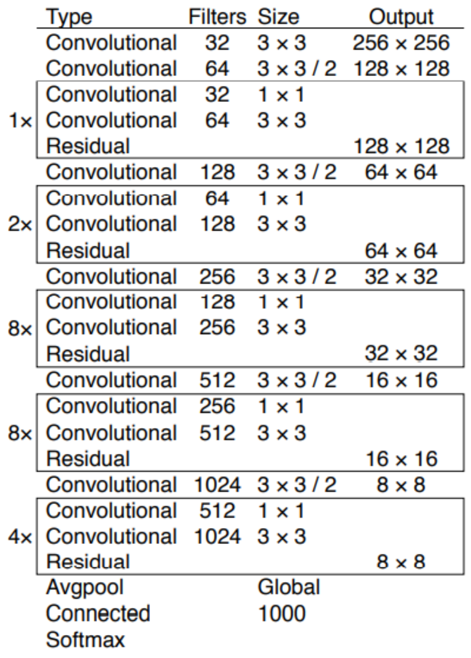
YOLOv3 employs a multilabel classification system. Instead of the SoftMax function, independent logistic classifiers are utilized here to decrease calculational complexity, which enhances system performance. An item may be labelled as both a fish and an underwater object in complex settings, such as when using an open Fish4Knowledge dataset, suggesting that there are many overlaying labels. SoftMax performs poorly since it forecasts the existence of a single class, which may not be the intended outcome; thus, binary cross entropy is employed in YOLOv3.

3.4.3 Predictions across scales

Three alternative scales are used to make predictions: 13x13, 26x26, and 52x52. The feature pyramid system is employed to gather characteristics, and then are processed by darknet53. The final phase of the forecast is a 3-D tensor that encodes the bounding box, the confidence rate which yields the objective score, and the likelihood that the item belongs to a specific class.

3.4.4 Feature extractor

For feature extraction, YOLOv3 employs the Darknet53 network, a fusion network resulting from Darknet19, and the residual network. This network has an aggregate of 53 convolutional layers and is known as darknet-53. Fig. 4 depicts the design of DarkNet53. For feature extraction, YOLOv2 uses darknet-19, while YOLOv3 employs darknet53 with 53 convolutional layers. Batch normalization is used in cooperation with YOLOv3 as well as YOLOv2.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
1×	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
2×	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
8×	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
8×	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
4×	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Darknet-53

Figure 4: Architecture of Darknet53

3.5 SSD MobileNetV2

The SSD image detection system combines detection and categorization into a system as in Fig. 5 to enable end-to-end training. The fundamental to training is to allocate the label data to one specific result in the predefined detector return collection. After this label is made, the function of loss is used from beginning to end to back-propagate to alter the weight of the network layer. The loss function used by the SSD detector is made up of regression as well as classification. The regression component that fails aspires to have as similar of a variance between the projected and default box gap and the ground truth as well as the default box as possible. This technique guides network learning by calculating the variance between the expected value and the ground truth value.

The MultiBox objective function is expanded by the SSD detector to include a loss function that can deal with several object types. The procedure is depicted in Fig. 7. Using the Jaccard coefficient, which calculated how similar each default box was to

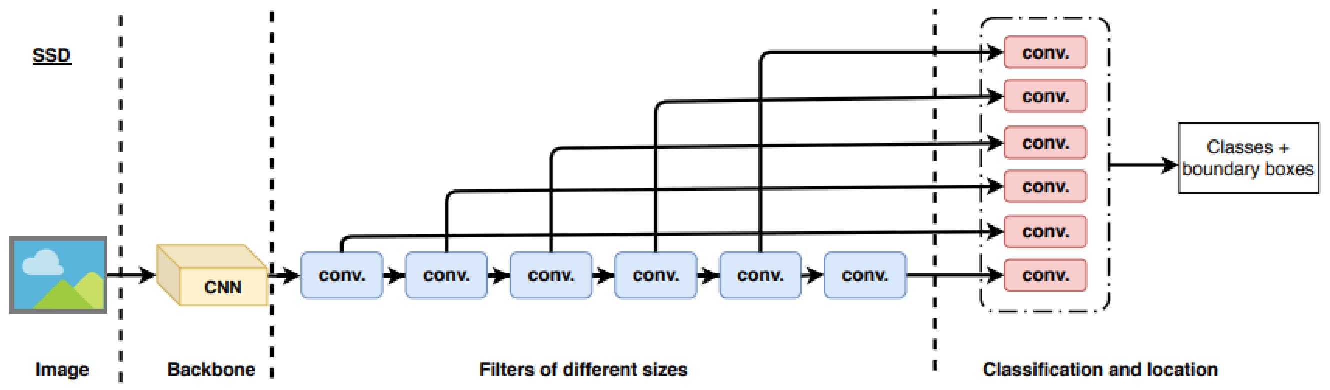


Figure 5: SSD Architecture

the ground truth box. The candidate default box can only be identified as the default box with a threshold superior to 0.5. The Jaccard coefficient of the i-th default box corresponding to the j-th ground truth box of class p is represented by $x_{i,j}^{p}$ if there exist Ns default boxes with a corresponding degree larger than 0.5. The value of $x_{i,j}^{p}$ is 0 if they do not match; else, it is 1. Numerous default boxes correspond to the j-th ground truth. box if the sum of $x_{i,j}^{p}$ is greater than or equal to 1. The weighted sum of the Lconf, as well as Lloc, is therefore the overall objective function, where is the weighting term for the Lloc. The Smooth L1 loss of the mismatch among the expected (l) as well as ground-truth box (g) is known as the Lloc the SSD ignores the negative example matching and solely penalizes the prediction box of the positive sample matching. According to the existing method, the Lconf is the prediction class's Sigmoid Focal loss. The class imbalance brought on by too much background can be corrected by this new classification loss. Then calculated the loss using the confidence score for the related class for every positive example matching prediction and the loss depending on the confidence score for the class "0" which cannot include the item for each negative example matching prediction. The Lconf plays a more important role in object detection for small and medium items than the Lloc. By using cross-validation, the SSD adjusts the weight value of the Lloc to 1 by default. The dataset's fish dimensions are not particularly big, though. As a result, it is changed the weight value from 0.8 to get rid of the missed detection phenomena.

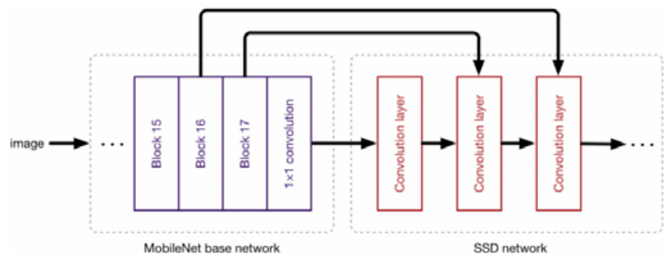


Figure 6: SSD MobileNetV2

3.5.1 Extracting multi-scale features with the MobileNetV2

Deeper and more sophisticated networks can be used to obtain higher accuracy. These networks do not, however, guarantee size and speed efficiency. For instance, the automatic feeding boat's fish identification operation in real-world applications must be implemented in real time on a constrained computer platform. As a result, it must build a multi-scale feature extraction system that is compact and quick as shown in Figure 6. To achieve this, the lightweight MobileNetV2 network as shown in Figure 7 is directly trained the highest factoring deconstruction capacity of the depth wise split convolution and the reversed residual topology with a strong memory inference method. Additionally, the prediction layer replaces normal convolution with depth wise separated convolution. This slightly lengthens the calculation period while guaranteeing that every extracted layer has the right determination as well as robust semantic characteristics.

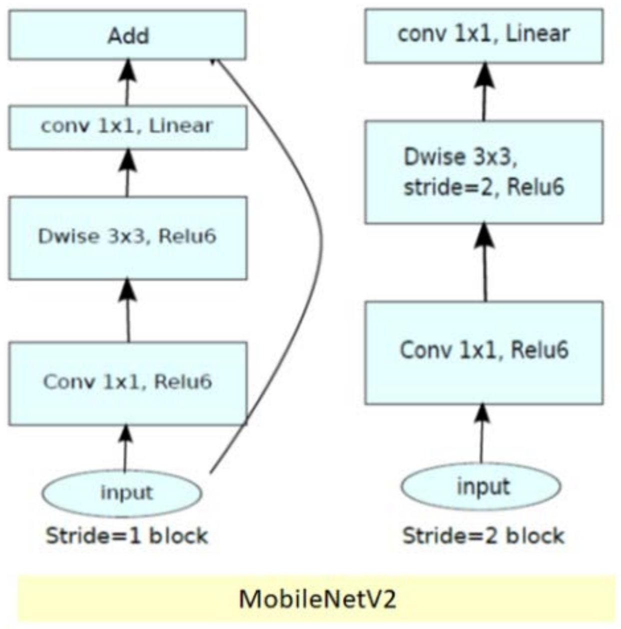


Figure 7: Mobile Net V2 Architecture

3.5.1.1. Depth-wise separable convolution

With depth-wise divisible convolution, the normal convolution process is split into two phases: filtering as well as combining. It is hypothesized that the N-channel $D_o * D_o$feature map would be produced after the M-channel $D_i * D_i$ input image had passed using a convolutional kernel of size $D_k * D_k$. Using N convolution kernels, the conventional standard convolution method continually extracts the source raw image's characteristics to create a result of N-channels, from which it is inevitable that many repeating characteristics will be retrieved. However, a single depth-wise separable convolution kernel can be used to extract features and produce multi-channel feature maps. Stage III's feature map addition is skipped if stages I and II are identical. Instead, filtering depth-wise convolution is used to directly stack the M-channel characteristic maps are overlaid with the one-channel characteristic maps.

The feature maps are then translated from the linear space of M dimensions to the space of N dimensions using a mixture of point-wise convolution, which uses N convolution kernels of size $1 * 1$. Depthwise differentiated convolution, as contrasted with regular convolution, utilizes N 1 * 1 convolution kernels for linear mapping and just one $D_k * D_k$ convolution kernel for feature extraction. In other words, they are reduced to $1/D_k^2 + 1/N$ of the normal convolution. The number of parameters is

$$M * N + M * D_k * D_k, \tag{5}$$

and there are methods for addition and multiplication is

$$D_k * D_k * D_o * D_o * M + D_o * D_o * M * N \tag{6}$$

With the 3 * 3 convolution kernel utilized in this study, it can reduce the number of functions and variables to around 1/9 to 1/8 of the classic convolution technique by only a small loss in accuracy. To eliminate parameter redundancy in depth-wise separable convolution, two essential hyper-parameters are also introduced. The feature maps' dimensions (number of channels) are dominated by the width multiplier [0, 1], while their resolution is dominated by the resolution multiplier. Thus, the depth-wise differentiated convolution's complexity in computation can be written as

$$D_k * D_k * \rho D_o * \rho D_o * \beta M + \rho D_o * \rho D_o * \beta M * \beta N \tag{7}$$

The network is successful in improving operational factors and minimizing redundant factors thanks to the interplay of these two hyper-parameters.

3.5.1.2 Inverted residual bottleneck structure

Due to the rectified linear unit's (ReLU) destruction of data properties, certain convolution kernels are vulnerable to training loss in real-world applications. ReLU processes on low dimensions can readily lead to data loss, while ReLU processes on high dimensions only lead to minimal data loss, according to the notion of manifolds of interest. To prevent data loss when employing ReLU for activation transformation in low-dimensional space, the author therefore thought about replacing the original non-linear activation transformation with a linear bottleneck structure. In addition, channel switching is not yet possible with depth-wise separable convolution. Because depth-wise segmented convolution is only able to filter spatial information in low-dimensional space channels per channel, the resultant characteristics are difficult to distinguish when there are limited channels of input. However, mapping channels, which can be utilized to increase as well as decrease their sizes, is a function of point-wise convolution. Using the lessons learned from the residual block, which is broad on both ends and narrower in the central area, a reversed residual bottleneck design was developed that is narrower on both ends and broad in the centre. To gradually expand the total amount of channels, point-wise convolution is first utilized; in this case, the expansion factor is set at 6, which is generally 5–11 times. Then, a small amount of depth-wise convolution processing was employed to filter the spatial data from a vast-channels, and then gather their characteristics. Point-by-point channel reduction was then accomplished by reusing the point-wise convolution. A low-dimensional linear bottleneck was created by this convolution using linear transformation. To reuse features, a shortcut structure like ResNet was included. This improved gradient propagation among multiplier layers and improved memory use efficacy. A fundamental component of MobileNetV2 is the designed depth-wise separable inverted residual bottleneck structure.

Additionally, the conventional non-linear activation function ReLU was enhanced by substituting ReLU6 for it, capping the highest output value at 6. By doing so, the section becomes further resilient in low-precision calculation and resolves the flaw that low-precision float16/int8 of portable systems cannot adequately express the accuracy loss produced by large-scale standards. As a conventional convolution kernel size, a $3 * 3$ convolution kernel was employed. A dropout mechanism is introduced to prevent over-fitting and a batch normalization technique to change the activation value distribution during training. Finally, MobileNetV2 performance was improved with minimal computational effort.

3.6 Faster R-CNN

The suggested underwater fish identification framework, Faster R-CNN, consists of binary modules. A deep fully convolutional network functions as the initial component, recommending areas, and a fast R-CNN detector functions as the next module, using the recommended areas. The whole system is a solitary, combined system for underwater fish identification (Figure 9). The RPN component tells the Fast R-CNN component where to look by using the recently popular terminology of neural systems with "attention" methods.

3.6.1 Region Proposal Networks

An image is input into a Region Proposal Network (RPN), which then generates several underwater item suggestions, every one of which has a rating for objectness. This section will address how a fully convolutional network is used to model this method. Since the ultimate objective is to exchange calculations with a Fast R-CNN underwater fish identification system, let us presume that the two nets use the same set of convolutional layers. For these studies, the Simonyan and Zisserman system (VGG-16) with 13 accessible convolutional layers and the Zeiler and Fergus system (ZF) with 5 accessible convolutional layers were both studied.

To create area proposals, slip a tiny network above the final shared convolutional layer's convolutional feature map. This tiny network receives a $n \times n$ spatial window of the input convolutional feature map as input. Every sliding window corresponds to a lower-dimensional feature (256-d for ZF and 512-d for VGG, respectively, with ReLU accordingly). This characteristic is input into two sibling fully connected layers, one for regression and one for classification. This research adopts n = 3 because the efficient receptive field on the input image is big (171 and 228 pixels for ZF and VGG, accordingly). Figure 8 depicts this mini-network in a single place. The fully connected layers are shared through each spatial place since the mini-network functions in a sliding window mode. This design is implemented naturally with a $n \times n$ convolutional layer after two sibling 1x1 convolutional layers which are cls and reg.

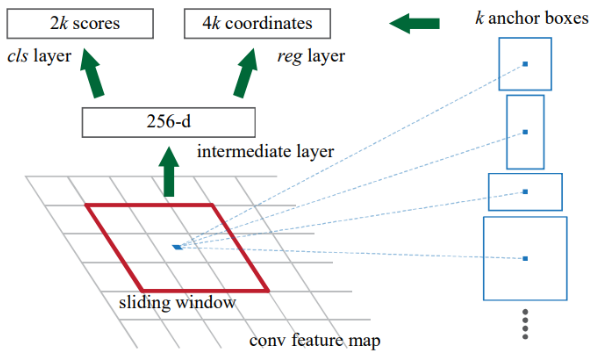


Figure 8: Region Proposal Network (RPN)

3.6.2 Anchors

It concurrently forecasts numerous area suggestions at every sliding-window location, where the amount of supreme feasible suggestions for every location is denoted as k. As an outcome, the reg layer has 4k results keeping the coordinates of k boxes, while the cls layer has 2k outputs estimating the likelihood of item or not item for every suggestion. The k suggestions are parameterized relative to k anchors, which are k reference boxes. A scale and aspect ratio are coupled with an anchor that is centered at the sliding window in question. It employs 3 scales and 3 feature proportions by default, producing $k = 9$ anchors at each sliding point. There is overall $W \times Hk$ anchors for a convolutional feature map of size $W \times H$ (generally 2,400).

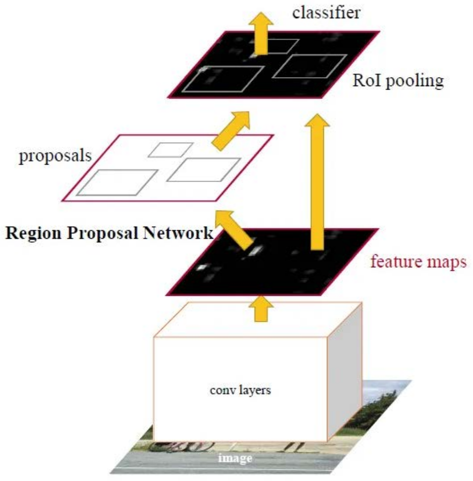


Figure 9: Faster R-CNN

3.6.3. Translation-Invariant Anchors

Translation invariance, in addition to regards to the computed functions and the anchor's suggestions comparative to the anchors, is a key characteristic of the proposed method. If an object in an image is translated, the proposal should translate as well, and the same function should be able to forecast the proposal in either place. The proposed method guarantees this translation-invariant property. In comparison, the MultiBox approach generates 800 anchors that are not translation invariant using k-means. As a result, MultiBox cannot assurance that the similar suggestion is produced when an item is transformed.

The model size is also decreased by the translation-invariant characteristic. When using k = 9 anchors, the proposed technique produces an output layer that is (4 + 2) 9-dimensional convolutional as opposed to MultiBox's (4 + 1) 800-dimensional fully-connected output layer. The output layer therefore has two orders of magnitude fewer parameters than MultiBox's output layer, which has 1536 (4 + 1) 800 for GoogleNet in MultiBox, or 2.8 104 parameters (512 (4 + 2) 9 for VGG-16). This suggestion layers nevertheless have a substantially smaller number of attributes than MultiBox even when the feature projection layers are taken into account. It is anticipated that the proposed approach will be less likely to overfit tiny datasets like PASCAL VOC.[34]

3.6.4. Regression References Using Multi-Scale Anchors

A new approach to dealing with various scales (and aspect ratios) is presented by the suggested design of anchors. There are 2 currently well-liked methods for multi-scale forecasts. The initial method depends on image/feature pyramids, such as in CNN-based algorithms as well as DPM. Multiple scales are used to resize the images, and for each scale, feature maps are generated. This method takes time but is frequently useful. The second method involves using sliding windows on the feature maps with various scales (and/or aspect ratios). For instance, DPM separately trains models with various aspect ratios using various filter sizes, such as 57 and 75. This technique can be observed as a "pyramid of filters" if it is applied to numerous scales. The next technique is typically utilized alongside the first method.

In contrast, the proposed anchor-based approach uses a pyramid of anchors, which is more economical. About anchor boxes with different sizes and aspect ratios, the proposed method categorizes and regresses bounding boxes. It only employs filters (slide windows on the feature map) of one size and only uses images and feature maps of one scale. This multi-scale design depending on anchors enables it to easily apply the convolutional features generated on a single-scale image, as is additionally performed by the Fast R-CNN detector. A crucial element for sharing features without incurring additional costs for addressing scales is the creation of multi-scale anchors.

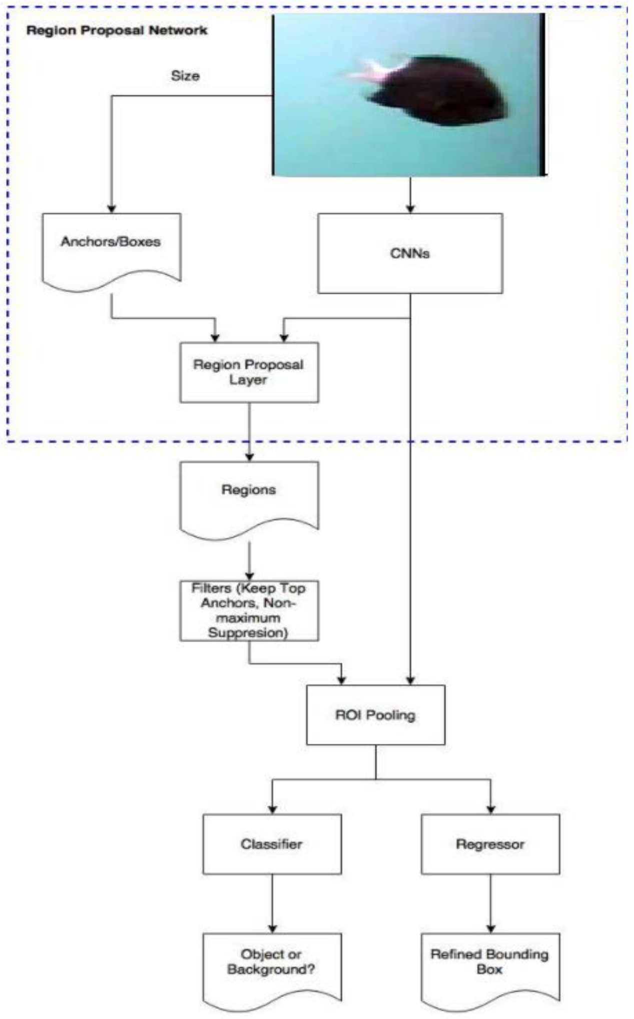


Figure 10: Faster R-CNN ResNet50

3.6.5 ResNet50 feature extraction:

The feature extraction network in the Faster R-CNN model is often a pre-trained CNN. The ResNet50 has been applied to it in this work as shown in Fig. 10. ResNet50 is a 50-layer, pre-trained CNN model. Using the Fish4Knowledge dataset, this model was trained on more than 2000 photos. Images can be categorized into different object categories. A 224x224 picture is used as the network's input.

ResNet50 has been used for the feature extraction step since the system has amassed extensive feature illustrations for a variety of images. Empirical analysis is the most effective method of selecting the best feature extraction layer. Stronger picture features are encoded by features that were retrieved further down the network.

The first 40 of the 46 depths in this investigation were employed using pre-trained ResNet50 to prove the benefits of the decrease further clearly. Two scenarios have been run to examine the impact of the ResNet50 depths on producing feature maps. In the initial example, the system reduces the dimensions of the 224x224x3 input image by 16 times. In the second case, the first 46 depths are used to reduce the input photos' dimension 32 times. After that, R-CNN was used to run both situations, and the outcomes were compared.

RESULT AND DISCUSSION

This segment describes the performance of the suggested method as well as the outcomes of its implementation. The results of the comparison with three methods such as YOLOv3, SSD MobileNetV2, and Faster R-CNN ResNet50 models proposed in this study are also shown.

4.1 Evaluation Indices

The Fish4Knowledge dataset was utilized to train and test the network. The record file created for the period of the training stage can be used to determine the losses in each batch. The loss and mean Average Precision (mAP) are displayed against iteration in Fig. 12 as well as 13. With each cycle, the loss lowers as well as the mAP growths. The system can be trained more up to the average loss falls below 0.2, at this point, the network becomes overfitted, which can be prevented by quitting early. The 3 recognition layers, layer 82, layer 94, as well as layer 106, compute the following loss coefficients for the bounding box: mean squared error of centerX, centerY, width, and height; binary cross entropy of objectness score, no objectness score, and multi-class forecasts. As a result, the loss function has 4 pieces and can be measured as follows.

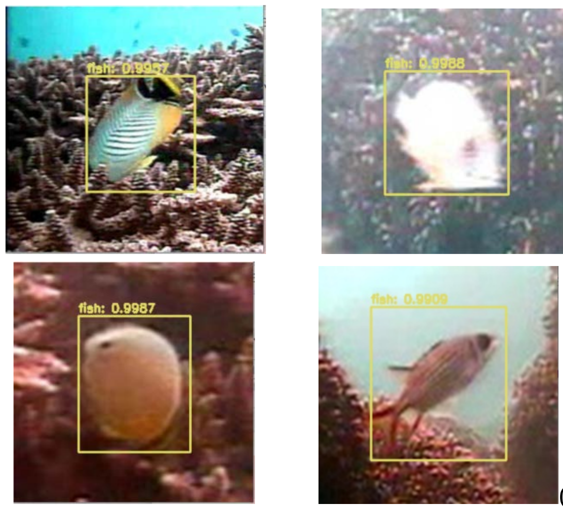
$$Loss = Lambda\_Coord * Sum(Mean\_Square\_Error((bx, by), (bx', by) * obj\_mask) + Lambda\_Coord * Sum(Mean\_Square\_Error((bw, bh), (bw >, bh') * obj\_mask) + Sum(Binary\_Cross\_Entropy(obj, obj') * obj\_mask) + Lambda\_Noobj * Sum(Binary\_Cross\_Entropy(obj, obj') * (1 - obj\_mask) * ignore\_mask) + Sum(Binary\_Cross\_Entropy(class, class')) \qquad (8)$$

Where bx and by indicate the relative centroid and bx' and by' represent the straight projected centroid. The weight Lambda_Coord has a value of 5. The next term represents the height and width loss obtained by combining width(bw) and height(bh), followed by the object/non-objectness score loss, and lastly the classification loss. The mAP was developed to evaluate the object detection performance as shown in Table 1. After 1600 iterations, the mAP was 98.51%, and an assurance level of 0.25 was established to evade obstruction of the bounding box. To acquire an enhanced outcome, mAP was evaluated with an IOU threshold of 0.5.

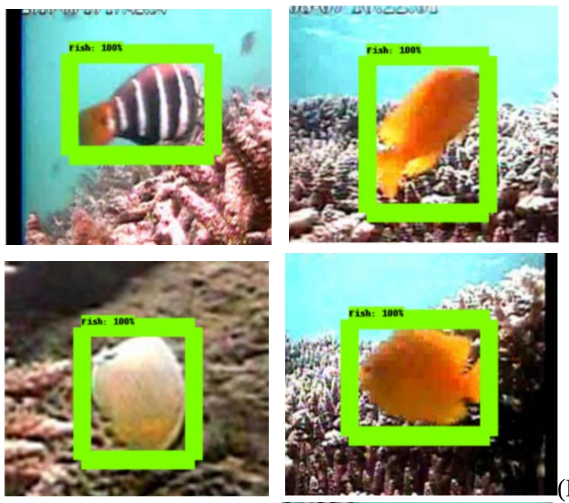
Both photos and videos were used to test the object detector. The outcomes attained by analysis the system by means of the Fish 4 Knowledge visual files of 09min 35sec period show that the best accuracy is 95.15% by using SSD MobileNetV2, the average loss is 0.475593, the precision is 0.9821, the recall is 0.9683, the F1 score is 0.9751, and the average IoU is 98.20, and the average detection time is 15 seconds, for confidence threshold = 0.25 and IoU threshold = 0.5. By considering the positive object class of fish in the background and the negative object class with a lack of fish in the background, the model's accuracy, precision, and recall are assessed. As indicated in (9), and (10), the mAP and IoU can be determined. The qualitative analysis of three models of YOLOv3, SSD MobileNetV2, and Faster R-CNN ResNet50 using the Fish4Knowledge dataset is shown in Fig. 11.

$$\text{mAP} = \frac{1}{No.class}\sum_{1}^{No\ of\ Class} \text{Average precision} \quad (9)$$

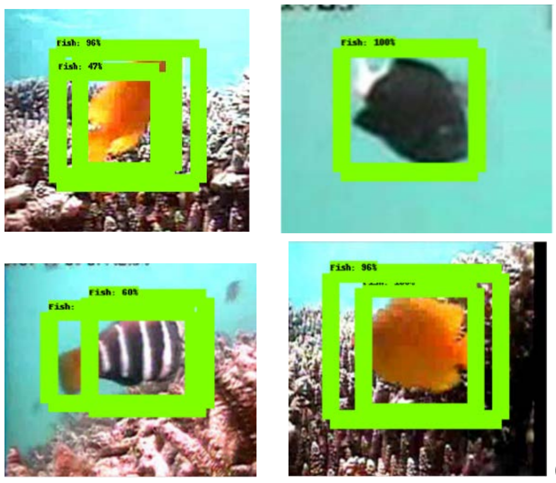
$$\text{IoU} = \frac{B1 \cap B2}{B1 \cup B2} \quad (10)$$



(A)



(B)

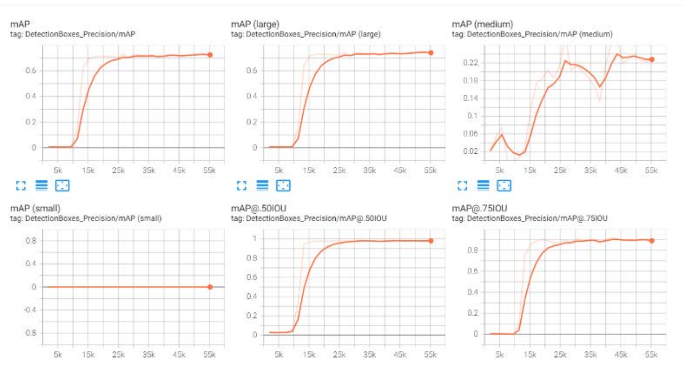


(C)

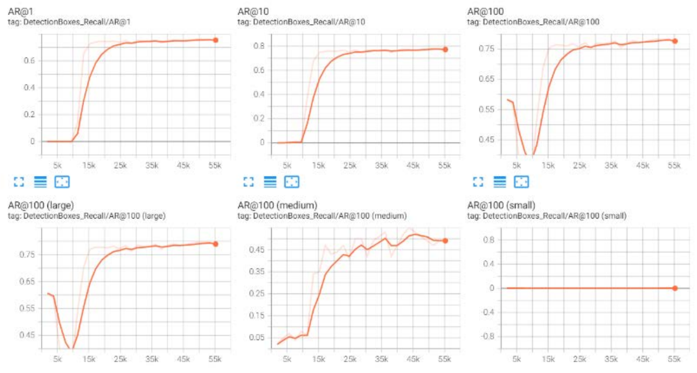
Figure 11: Qualitative Analysis of Models (A) YOLOv3 (B) SSD MobileNetV2 (C) Faster R-CNN ResNet50

Table 1: Comparison of mAP (mean Average Precision)

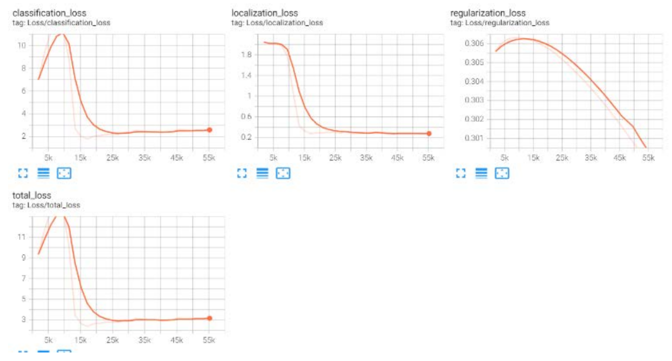
Models	IoU	Area	MaxDets	Average Precision (AP)
	0.50:0.95	all	100	0.718
SSD MobileNetV2	0.50	all	100	0.982
	0.75	all	100	0.876
	0.50:0.95	all	100	0.583
Faster R-CNN ResNet50	0.50	all	100	0.926
	0.75	all	100	0.696
	0.50:0.95	all	100	0.681
YOLOv3	0.50	all	100	0.985
	0.75	all	100	0.820



(a)



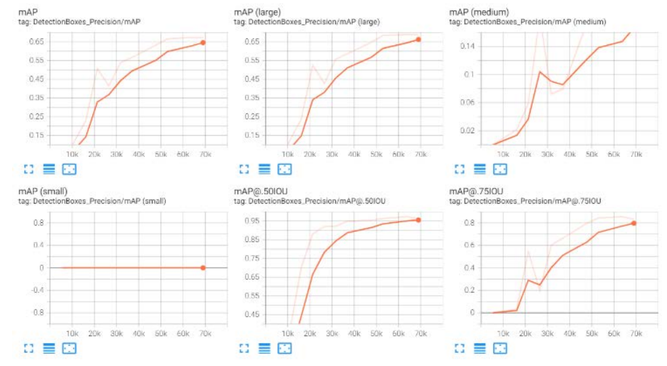
(b)



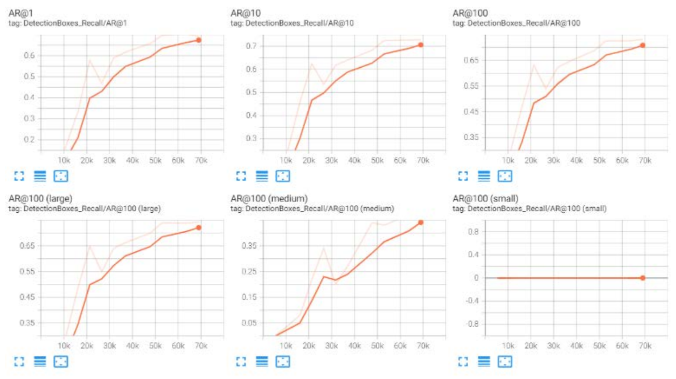
(c)

Figure 12: SSD MobileNetV2 (a) Precision (b) Recall (c) Loss

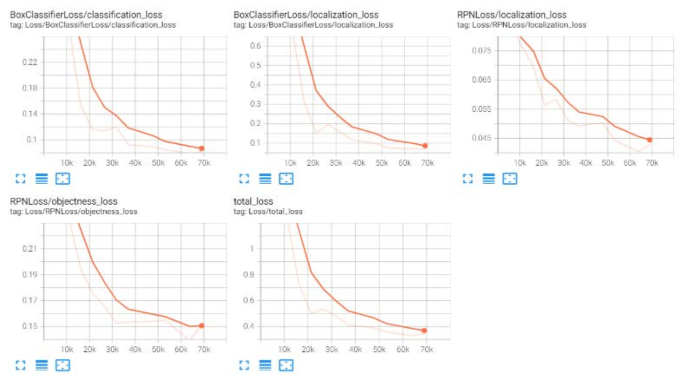
From Figure 12, the precision, recall, and loss of the proposed SSD MobileNetV2 model are shown under various conditions. The precision is calculated based on the mean average precision under various IOU values of the decision box such as small, large, medium, 50IOU, and 75IOU. Thus, the SSD MobileNetV2 model achieves a better precision of above 98% for 50IOU and a lower precision of 0% for small IOU values. Then the recall is calculated based on the different AR values such as 1, 10, 100, and 100 (small, large, and medium). Thus, the SSD MobileNetV2 model obtained a higher recall during the AR value is 10 and the lowest recall at 100 (small) AR value. Similarly, the loss is calculated using different types of loss such as classification loss, localization loss, regularization loss, and total loss. Thus, the SSD MobileNetV2 model achieves the lowest loss in terms of regularization loss.



(a)



(b)



(c)

Figure 13: Faster R-CNN ResNet50 (a) Precision (b) Recall (c) Loss

From Figure 13, the precision, recall, and loss of the proposed Faster R-CNN ResNet50 model is shown under various conditions. The precision is calculated based on the mean average precision under various IOU values of the decision box such as small, large, medium, 50IOU, and 75IOU. Thus, the Faster R-CNN ResNet50 achieves a better precision of above 95% for 50IOU and a lower precision of 0% for small IOU values. Then the recall is calculated based on the different AR values such as 1, 10, 100, and 100 (small, large, and medium). Thus, the Faster R-CNN ResNet50 obtained a higher recall during the AR value is 10 and the lowest recall at 100 (small) AR value. Similarly, the loss is calculated using different types of loss such as classification loss, localization loss, RPN loss, objectness loss, and total loss. Thus, the Faster R-CNN ResNet50 achieves the lowest loss in terms of localization loss _ RPN loss.

Accuracy is used to gauge how well the categorization model performs. It also refers to the percentage of reliable results (TP or TN). ACC is often acquired through

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \tag{11}$$

Wherein TP, FP, TN, as well as FN, denotes "true positives," "false positives," "true negatives," as well as "false negatives," accordingly.

The proportion of confirmed positive cases among all expected positive patterns is known as precision. The formula below can be used to compute it.

$$Precision = \frac{TP}{TP+FP} \tag{12}$$

The percentage of true positives received by a classifier is used to calculate recall, which indicates how successfully it can recognize positive class patterns. The process might be sensitive without being, or exact without being susceptible. The recall could be calculated using the formula

$$Recall = \frac{TP}{TP+FN} \tag{13}$$

F1-score is a more accurate predictor of incorrectly detected patterns than ACC. It is determined by using 10

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{14}$$

Table 2: Performance Estimation of the suggested models as YOLOv3, SSD MobileNetV2, Faster R-CNN ResNet50

Parameters	YOLOv3	SSD MobileNetV2	Faster R-CNN ResNet50
Recall	**0.99**	**0.9683**	**0.9647**
Precision	**0.79**	**0.9821**	**0.4506**
F-Score	**0.88**	**0.9751**	**0.6142**
Accuracy	**0.78**	**0.9515**	**0.4433**

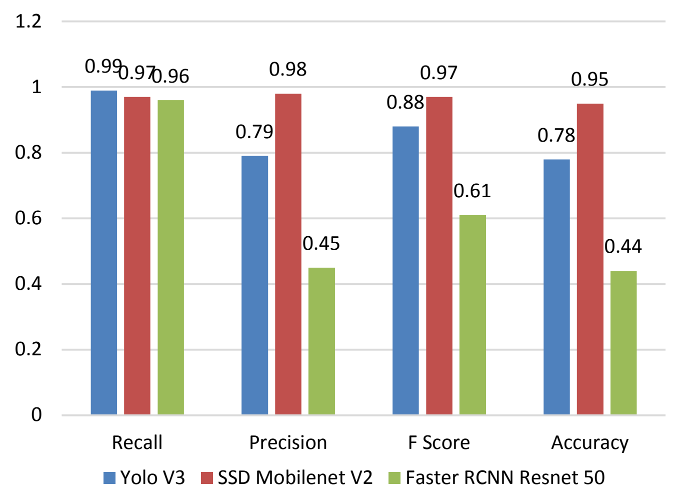


Figure 14: Performance Evaluation of the proposed models as YOLOv3, SSD MobileNetV2, Faster R-CNN ResNet50

From Table 2, the proposed model-based performance metrics are evaluated using Accuracy, Precision, F1-Score, and Recall. Thus, the performance parameters of the models have the values of Accuracy 0.78, Precision 0.79, Recall 0.99, and F1-Score 0.88 for the YOLOv3 model, Accuracy 0.9515, Precision 0.9821, Recall 0.9683, as well as F1-Score, is 0.9751 for the SSD MobileNetV2 model, and Accuracy is 0.4433, Precision is 0.4506, Recall is 0.9647 as well as F1-Score is 0.6142 for the Faster R-CNN ResNet50 model as depicted in Fig. 14.

4.2. Comparison

The following section provides the comparison outcomes regarding the precision, recall, as well as F-measure of the proposed method. The seven models such as MFI [35], MOG2, LSBP, Hybrid of MOG2 and LSBP, YOLOv3 (proposed), SSD MobileNetV2 (proposed), and Faster R-CNN ResNet50 (proposed) are using the Fish4Knowledge dataset which is equated on account of precision, recall and F-measure and the comparison results shows that the implemented SSD MobileNetV2 achieves best results as the recall is 0.9683, precision is 0.9821 and F-Measure is 0.9751 as depicted in Table:3.

Table 3: Comparison of proposed method based on Performance Metrics

Sr. No.	Performance metric Parameters	MFI [35]	MOG2	LSBP	Hybrid of MOG2 and LSBP	YOLOv3	SSD Mobile Net V2	Faster R-CNN ResNet50
1	Recall	0.57	0.58	0.54	0.611	0.99	0.9683	0.9647
2	Precision	0.73	0.97	0.95	0.73	0.79	0.9821	0.4506
3	F-Measure	0.64	0.72	0.68	0.64	0.88	0.9751	0.6142

With real-time processing, multi-scale detection, reduced latency, object bounding boxes, and lighting situation adaptability, SSD MobileNetV2 is a compact and effective underwater fish detection system compared to YOLOv3 (and Faster R-CNN ResNet50. Its multi-scale detection skills enable it to recognize fish of varied sizes and orientations inside the same frame, and its lightweight architecture allows it to run effectively on autonomous underwater vehicles and fixed camera setups. Compared to multi-stage methods, its single-shot architecture helps achieve lower latency. Users can effectively observe fish behaviour and interactions in dynamic underwater habitats by carefully regulating variables such as dataset quality, fine-tuning, and accuracy trade-offs.

CONCLUSION

Finally, using the unique dataset Fish4knowledge, this work investigated the effectiveness of YOLOv3, SSD MobileNetV2, and Faster R-CNN ResNet50 models for object detection in dynamic underwater backgrounds. The models' capabilities were assessed using performance indicators. The trial results demonstrated that the SSD MobileNetV2 method performed better than the YOLOv3 as well as Faster R-CNN ResNet50 models in terms of precision due to the properties of SSD MobileNetV2 such as real-time processing, multi-scale detection, reduced latency, object bounding boxes, and lighting situation adaptability. When used to the Fish4knowledge dataset, this model produced promising results when compared to other cutting-edge approaches. SSD MobileNetV2's effectiveness in underwater fish detection demonstrates its real-world potential in marine biology research, environmental monitoring, and autonomous underwater systems. Continuous research and developments in deep learning methodologies will improve underwater fish detection techniques and lead to better knowledge and protection of underwater ecosystems.

REFERENCES

1. G. M. Vianna, D. Zeller, & D. Pauly, Fisheries and policy implications for human nutrition. *Current Environmental Health Reports* **2020**, 7, 161-169.
2. H. Blemel, A. Bennett, S. Hughes, K. Wienhold, T. Flanigan, M. Lutcavage, & C. Tam. Improved fish tagging technology: field test results and analysis. *In OCEANS 2019-Marseille* **2019**, 1-6.
3. R. Hilborn, R. O. Amoroso, C. M. Anderson, J. K. Baum, T. A. Branch, C. Costello, & Y. Ye, Effective fisheries management instrumental in improving fish stock status. *Proceedings of the National Academy of Sciences* **2020**, 117(4), 2218-2224.
4. B. V. Deep, & R. Dash. Underwater fish species recognition using deep learning techniques. *In 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)* **2019**, 665-669.
5. S.N. Kakarwal, A.S. Gavali. Context aware human activity prediction in videos using Hand-Centric features and Dynamic programming based prediction algorithm. *J. Integr. Sci. Technol.* **2022**, 10 (1), 11–17.
6. Y. LeCun, Y. Bengio, & G. Hinton. Deep learning. *Nature* **2015**, 521(7553), 436-444.
7. F. Capoccioni, C. Leone, D. Pulcini, M. Cecchetti, A. Rossi, & E. Ciccotti. Fish movements and schooling behavior across the tidal channel in a Mediterranean coastal lagoon: An automated approach using acoustic imaging. *Fisheries Research* **2019**, 219, 105318.
8. A. Saleh, M. Sheaves, & M. Rahimi Azghadi. Computer vision and deep learning for fish classification in underwater habitats: A survey. *Fish and Fisheries* **2022**, 23(4), 977-999.

9. J. G. A. Barbedo. A review on the use of computer vision and artificial intelligence for fish recognition, monitoring, and management. *Fishes* **2022**, 7(6), 335.
10. A. Olsen, D. A. Konovalov, B. Philippa, P. Ridd, J. C. Wood, J. Johns, & R. D. White. DeepWeeds: A multiclass weed species image dataset for deep learning. *Scientific reports* **2019**, 9(1), 2058.
11. M. R. Azghadi, C. Lammie, J. K. Eshraghian, M. Payvand, E. Donati, B. Linares-Barranco, & G. Indiveri. Hardware implementation of deep network accelerators towards healthcare and biomedical applications. *IEEE Transactions on Biomedical Circuits and Systems* **2020**, 14, 1138–1159.
12. A. Saleh, I. H. Laradji, C. Lammie, D. Vazquez, C. A. Flavell, & M. R. Azghadi. A deep learning localization method for measuring abdominal muscle dimensions in ultrasound images. *IEEE Journal of Biomedical and Health Informatics* **2021**, 25, 3865–3873.
13. R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, & B. Hodjat. Evolving Deep Neural Networks. *In Artificial Intelligence in the Age of Neural Networks and Brain Computing* **2019**, 293–312.
14. G. Montavon, W. Samek, & K. R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* **2018**, 73, 1–15.
15. H. Zheng, R. Wang, W. Ji, M. Zong, W. K. Wong, Z. Lai, & H. Lv. Discriminative deep multi-task learning for facial expression recognition. *Information Sciences* **2020**, 533, 60–71.
16. P. Saini, K. Kumar, S. Kashid, A. Saini, & A. Negi. Video summarization using deep learning techniques: a detailed analysis and investigation. *Artificial Intelligence Review* 1-39.
17. N. F. F. Alshdaifat, A. Z. Talib, & M. A. Osman. Improved deep learning framework for fish segmentation in underwater videos. *Ecological Informatics* **2020**, 59, 101121.
18. D. An, J. Hao, Y. Wei, Y. Wang, & X. Yu. Application of computer vision in fish intelligent feeding system—A review. *Aquaculture Research* **2021**, 52(2), 423-437.
19. W. Rawat, & Z. Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation* **2017**, 29(9), 2352-2449.
20. H. J. Yoo. Deep convolution neural networks in computer vision: a review. *IEEE Transactions on Smart Processing & Computing* **2015**, 4(1), 35-43.
21. Y. Wang, C. Wang, H. Zhang, Y. Dong, & S. Wei. Automatic ship detection based on RetinaNet using multi-resolution Gaofen-3 imagery. *Remote Sensing* **2019**, 11(5), 531.
22. Y. Wang, C. Wang, & H. Zhang. Combining a single-shot multi-box detector with transfer learning for ship detection using sentinel-1 SAR images. *Remote sensing letters* **2018**, 9(8), 780-788.
23. K. Cai, X. Miao, W. Wang, H. Pang, Y. Liu, & J. Song. A modified YOLOv3 model for fish detection based on MobileNetv1 as backbone. *Aquacultural Engineering* **2020**, 91, 102117.
24. M. Sung, S. C. Yu, & Y. Girdhar. Vision-based real-time fish detection using convolutional neural network. *In OCEANS 2017-Aberdeen* **2017**, 1-6.
25. S. Cao, D. Zhao, X. Liu, & Y. Sun. Real-time robust detector for underwater live crabs based on deep learning. *Computers and Electronics in Agriculture* **2020**, 172, 105339.
26. G. Liu, J. C. Nouaze, P. L. Touko Mbouembe, & J. H. Kim. YOLO-tomato: A robust algorithm for tomato detection based on YOLOv3. *Sensors* **2020**, 20(7), 2145.
27. H. Zhao, Y. Zhou, L. Zhang, Y. Peng, X. Hu, H. Peng, & X. Cai. Mixed YOLOv3-LITE: A lightweight real-time object detection method. *Sensors* **2020**, 20(7), 1861.
28. B.S. Panda, K.M. Gopal, R.N. Satapathy. RRBCNN: Aircraft detection and classification using Bounding Box Regressor based on Scale Reduction module. *J. Integr. Sci. Technol.* **2023**, 11 (1), 412.
29. K. M. Knausgård, A. Wiklund, T. K. Sørdalen, K. T. Halvorsen, A. R. Kleiven, L. Jiao, & M. Goodwin. Temperate fish detection and classification: a deep learning based approach. *Applied Intelligence* **2022**, 1-14.
30. A. Jalal, A. Salman, A. Mian, M. Shortis, & F. Shafait. Fish detection and species classification in underwater environments using deep learning with temporal information. *Ecological Informatics* **2020**, 57, 101088.
31. A. Testolin, D. Kipnis, & R. Diamant. Detecting submerged objects using active acoustics and deep neural networks: A test case for pelagic fish. *IEEE Transactions on Mobile Computing* **2020**, 21(8), 2776-2788.
32. H. S. Chhabra, A. K. Srivastava, & R. Nijhawan. A hybrid deep learning approach for automatic fish classification. *In Proceedings of ICETIT 2019: Emerging Trends in Information Technology* **2020**, 427-436.
33. Y. Han, Q. Chang, S. Ding, M. Gao, B. Zhang, & S. Li. Research on multiple jellyfish classification and detection based on deep learning. *Multimedia Tools and Applications* **2022**, 1-16.
34. Redmon, Joseph and Farhadi, Ali. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* **2018**.
35. M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, & A. Zisserman. The Pascal visual object classes challenge: A retrospective. *International journal of computer vision* **2015**, 111, 98-136.
36. S. Vasamsetti, S. Setia, N. Mittal, H. K. Sardana, & G. Babbar. Automatic underwater moving object detection using a multi-feature integration framework in complex backgrounds. *IET Computer Vision* **2018**, 12(6), 770-778.