Fast and Accurate Fish Detection Design with Improved YOLO-v3 Model and Transfer Learning

Kazim Raza[1], Song Hong[2]
Department of Ocean Science and Engineering
Ocean Opto, Electronics and Automation Lab
Zhejiang University, Hangzhou Zhejiang 310058, China

***Abstract*—Object Detection is one of the problematic Computer Vision (CV) problems with countless applications. We proposed a real-time object detection algorithm based on Improved You Only Look Once version 3 (YOLOv3) for detecting fish. The demand for monitoring the marine ecosystem is increasing day by day for a vigorous automated system, which has been beneficial for all of the researchers in order to collect information about marine life. This proposed work mainly approached the CV technique to detect and classify marine life. In this paper, we proposed improved YOLOv3 by increasing detection scale from 3 to 4, apply k-means clustering to increase the anchor boxes, novel transfer learning technique, and improvement in loss function to improve the model performance. We performed object detection on four fish species custom datasets by applying YOLOv3 architecture. We got 87.56% mean Average Precision (mAP). Moreover, comparing to the experimental analysis of the original YOLOv3 model with the improved one, we observed the mAP increased from 87.17% to 91.30. It showed that improved version outperforms than the original YOLOv3 model.**

Keywords*—Deep learning; computer vision; transfer learning; improved YOLOv3; anchor box; custom dataset

I. Introduction

Deep learning (DL) is the subfield of Machine learning (ML), which is built on artificial neural networks that can be unsupervised, semi-supervised, or supervised learning. The methods of DL are characterization learning methods that acquired from nonlinear modules to transform raw data representation into a higher level. The core aspect of DL is that layers acquired from the given data, unlike humans [1]. Researchers tried hard to train a deep multi-layer network for decades, but still, before 2006, there were not many successful experiments at that time where they only passed on effective results with one or two hidden layers. Those results were not producing substantial outcomes due to exploding gradients. DL is like a sensory system where the flow of information having internal connections with all of the neurons, and every neuron helps to process the information to the next one.

There is a massive difference between DL and ML, ML only relies on structured data, whereas DL required layers of the Artificial Neural networks. Szeged at el [2] Deformable Part Model (DPM) is one of the top techniques for object recognition that's implementation is established on the decomposition of the object and expressed in graphical mode. This model has only two layers that are not useful for the big dataset. Traditional ML classifiers likewise SVM, LDA, which is insufficient for huge dataset classification. The hierarchical Classification is quite exceptional than SVM because of its 4% accuracy results than a flat SVM classifier [3]. In the previous traditional methods, the researches never used the deep Convolutional Neural Network (CNN) design as they were using a tiny dataset that has a low range of images and a restricted number of fish species. Another critical point to remember, they were using handcrafted ways; that is why performance was not up to the mark. The implemented algorithm was inadequate for a big dataset, and resultantly the accuracy not achieved consequently. In the recent past, the Fast R-CNN, and faster R-CNN gain significant research performance, but these architectures have a very complex execution pipeline to perform recognition tasks. These architectures have less Frame Per Second (FPS) and accuracy as well. We proposed the YOLOv3 real time object detection model in our research work.

The major contribution of this work is given as follows:

- We improved the model by the addition of a 4th detection scale in the network to enhance the performance by obtaining finer-grained features.
- Applied K-means++ clustering on our dataset to get suitable anchor boxes and increase the anchor boxes from 9 to 12.
- Applied a novel transfer learning method to improve efficiency.
- We also changed the loss function for learning and convergence in the model.

The rest of the paper is described as follows:

Section II explains the background study. Section III explains the research methodology, including improvements in the methodology. Section IV explains the dataset composition and its structure. Section V explains the results and comparison of different state of the art object detectors. In the end, Section VI explains the conclusion and future direction.

II. Background Study

In the deep ocean, the movement of the fish is unpredictably quick and three-dimensionally; therefore, recognition is a difficult task. Fish recognition depicts to identify different types of fish species according to their features. It is essential to locate for other kinds of reasons,

including contour and pattern matching, statistical, quality control, feature extraction, and determination of physical traits [4]. Larsen at el. [5] obtained the shape and texture feature from appearance model and testing on the dataset, which has been containing more than 100 images of three fish species and attaining the accuracy 76%. Helge Balk at el. [6] developed the Sonar5 post-processing program that covered interpretation, analysis, and acquisition stages of hydroacoustic fish detection. The fish-echoes, along with surrounding noise level, can be detected using this program due to its time variation in sonar's detection, so the overall accuracy was high. Fuming Xiang at el. [7] used CNN models pipeline, including VGG16 and SSD on 9 common species of fish in the Missouri river to classify into category and position. They have achieved 87.22% accuracy in the classification of the fish.

Recognizing fish is one of the possibilities that come out with DL, which helps to find the targeted underwater species, i.e., fish. There are hundreds of applications to recognize marine fish, and many practices have already been done to find the right one object, which helps people to solve the problem. Tracking and counting the fish is also crucial for fish industry and conservation purposes as well.

The exact quantity of slaughtering fish is not final yet. Still, there is an estimated figure that salmon, sea trout, and migratory char are 27.0% decreased in killing fish from 2017 to 2018, according to Statistics Norway [8]. As per the report, the global river catch has passed to almost 10 million right after the linear growth from the 1950s, which was under-reported on collecting the relevant data in the past [9]. There is no certainty on how much river fish caught, released, or slaughtered after catching from the river or ocean, so this thing needs some automation with an accuracy of data.

Moreover, the caught fish is healthy or not needs some consideration and observation to determine whether the fish is healthy as not all fishes can be healthy.

For all of such problems, the CNN does help in the classification of the marine system, observing the behavior of the underwater object, tracking an accurate object, automated, accurate counting of fish caught globally, localization, and controlling the environment.

There almost 20 deep neural networks have been trained for Salmon fish recognition that provides an in-depth discussion of each model with parameter tuning [10]. Moreover, SSD version 2 achieved 84.64% mAP, state-of-the-art accuracy with 3.75 FPS for salmon recognition. Background subtraction method used to detect and track fish in marine life with the help of a video sequence. They get an accurate 73% result from the real type of video though they get the best result by implementing the Viola-Jones method using Haar cascade [11].

Undoubtedly, fish recognition is a complex task where some of the challenges like noise, distortion, overlap, occlusion, and segmentation error needs several techniques to get some accuracy in the result. Some of the techniques have already applied, and one of the SVM based techniques used on the two training sets on the fish features [12]. One was containing 74 fish testing set, and the other was about 76 fish. The final result based on SVM showed 78.59% accuracy in the fish classification. Dhruv Rathi et al. [13] derived a method based on CNN for the automation classification of fish species, which achieved 96.29% accuracy than other proposed systems.

III. Research Method

Object recognition and detection are important issues in CV problems. Based on the detection pipeline and backbone architecture, the object detector algorithm classified into two types (1) two-stage object detectors such as fast R-CNN [14], faster R-CNN [15], Mask R-CNN [16], and (2) single-stage object detectors such as SSD [17], YOLO [18], YOLOv2 [19], YOLOv3[20]. The two-stage detection algorithm computationally very complex because they have separate backbone architecture. The single-stage object detector models are computationally less complex than that of the two-stage detector. The single stage detection algorithm like YOLOv3 is much faster, and the accuracy of YOLOv3 and faster R-CNN have no larger difference. So we implement the YOLOv3 object detection model in this paper, which is a fast and real- time object detection model. For the feature extraction, YOLOv3 use darknet-53 as a backbone architecture. The first and second versions of YOLOv3 architecture struggle with small object recognition. As we detect fishes so this 53 convolutional layers' architecture for feature extractor is the best choice. The backbone architecture of YOLOv3 still performs better than ResNet-101 and ResNet-152.

The backbone darknet-53 holds 23 residual units, and every such unit performs the 1×1 and 3×3 convolutional. At the end of every residual unit, an element-wise addition carried out between the input and output vectors. Every convolutional layer pursued by the Leaky ReLU activation function, where Batch Normalization is using. The downsampling runs with a stride of 2 at five separate convolutional layers.

YOLOv3 implements a Feature Pyramid Network (FPN) that used to detect the objects at different scales that constructs FPN on top of backbone architecture and build a pyramid with downsampling strides, 8, 16, and 32 in order to detect all-sized objects. The improved network structure of Darknet-53 shown in Fig. 1. We proposed the 4th scale to increase the detection performance where the red box represents the 4th detection scale, which helps to increase the detection of extra small objects with the downsampling stride of 4×. It helps us to get more exceptional grained features to detect extra small size targeted objects. The experimental scheme of the proposed work is shown in Fig. 2, which illustrates the initial dataset collection, pre-processing, and labeling of the dataset. Then we applied transfer learning on our custom dataset and fine-tuned the model to get better results on the custom dataset. We trained our model as much as it is converged and finally checked the visualization detection results and evaluation of the model.

	Layers	Filters	Strides	Output
	Convolutional	32	3×3	416×416
	Convolutional	64	3×3/2	208×208
1×	Convolutional	32	1×1	
	Convolutional	64	3×3	
	Residual			208×208
	Convolutional	128	3×3/2	104×104
2×	Convolutional	64	1×1	
	Convolutional	128	3×3	
	Residual			104×104
	Convolutional	256	3×3/2	52×52
8×	Convolutional	128	1×1	
	Convolutional	256	3×3	
	Residual			52×52
	Convolutional	512	3×3/2	26×26
8×	Convolutional	256	1×1	
	Convolutional	512	3×3	
	Residual			26×26
	Convolutional	1024	3×3/2	13×13
4×	Convolutional	512	1×1	
	Convolutional	1024	3×3	
	Residual			13×13

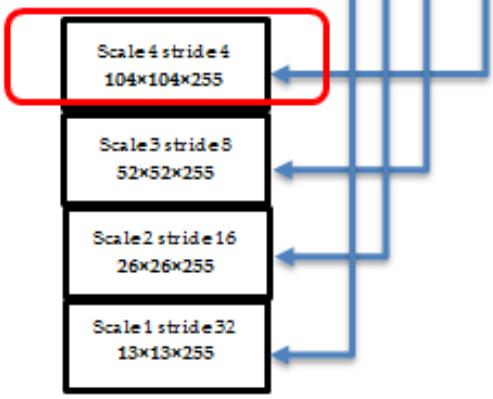


Fig. 1. Improved Network Structure of YOLOv3.

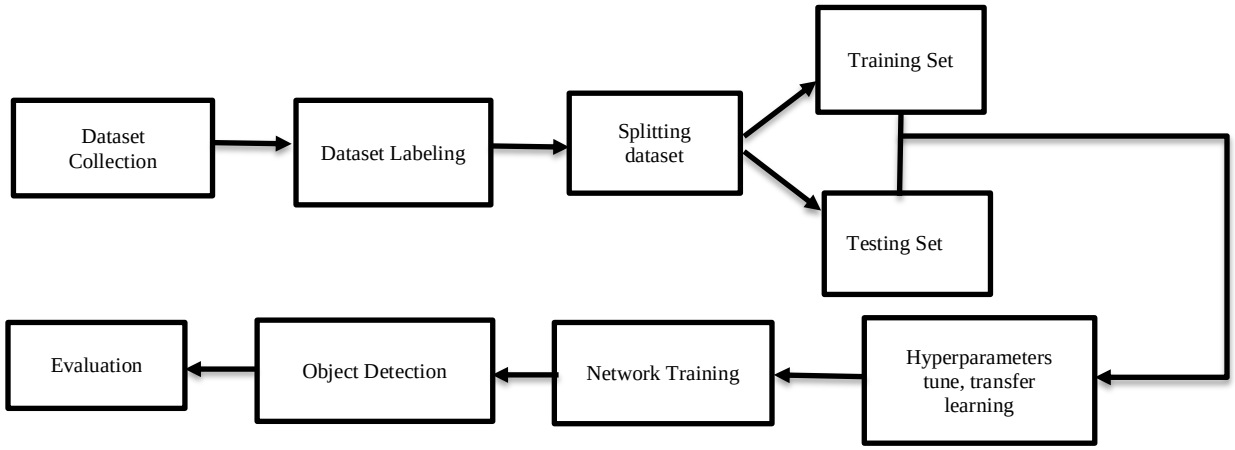


Fig. 2. Dataset Management and Detection Flow Diagram.

A. Algorithm Implementation Parameters

In improved YOLOv3, we have changed several default parameters to make the algorithm more robust. Unlike YOLOv3 in improved YOLOv3, we enhance the FPN because of increasing detection scales. Addition of 3 types of data augmentation in the algorithm for better training and testing results, improvement in the loss function, increase in anchor boxes, configuring of the tensor board to visualize the entire network performance. The algorithm parameters are shown in Table I.

B. K-means ++ Clustering

YOLOv3 used the idea of anchor boxes during the prediction of a bounding box. We increased the detection scale from 3 to 4 and used a custom dataset. With these effects in the network model, we ran a k- means++ clustering algorithm on our dataset to get the suitable size of anchor boxes for more improvement in detection accuracy. Besides, we increased the anchor boxes from 9 to 12 because we increased the detection scales from 3 to 4. Assign 3 anchor boxes to each detection scale depending upon the size of the object. The 12 anchor boxes generated by running the k-means++ clustering on our dataset are: (38, 23), (78, 52), (112, 84), (127, 117), (194, 98), (165, 139), (243, 155), (199, 205), (297, 237), (302, 280), (286, 343), (318, 374).

C. Improved YOLOv3 Loss Function

In the original paper of YOLOv3, the author used logistic regression to predict an objectness score for each bounding box that calculated the cost function. The objectness score is 1 if the anchor box overlaps the ground truth by more than or equal to a specific threshold value. On the other hand, if it still overlaps ground truth by less threshold value, that will not be considered the best bounding box. In Equation (1), we can see how the network output is changed by bounding box predictions where coordinates tx, ty, tw, th are responsible for computing the prediction.

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y \quad (1)$$
$$b_w = p_w e^{tw}$$
$$b_h = p_h e^{th}$$

The loss function is responsible for calculating the error between the real values and predicted one. The YOLOv3 loss function is the total sum of the coordinate loss, class loss, and confidence loss defined in equations (2), (3), and (4).

$$\text{Loss}_{\text{coord}} = \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[\left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2 \right],$$
$$+\lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[\left(\frac{w_i - \widehat{w_i}}{\widehat{w_i}}\right)^2 + \left(\frac{h_i - \widehat{h_i}}{\widehat{h_i}}\right)^2 \right] \quad (2)$$

$$\text{Conf}_{\text{loss}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left(C_i - \widehat{C}_i\right)^2$$
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} \left(C_i - \widehat{C}_i\right)^2 \quad (3)$$

$$\text{Class}_{\text{loss}} = \sum_{i=0}^{S^2} 1_i^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2 \quad (4)$$

Above loss function, x_i, y_i are the center coordinates of the ith box grid cell. h_i, w_i are the height and width and height of the i-th grid cell, respectively x_i, y_i, w_i and h_i are the real value and $\hat{x}_i$, $\hat{y}_i$, $\widehat{w_i}$ and $\widehat{h_i}$ are the predicted values. ($p_i(c)$ is the probability of a class and ($p_i(\widehat{c})$ is the corresponding prediction value. λ_{coord} coordinate loss of weights and λ_{noobj} is the bounding box object loss without weights. 1_{ii}^{obj} denotes that the j-th box predictor in cell i is "responsible" for that prediction. We used $\left(\frac{w_i - \widehat{w_i}}{\widehat{w_i}}\right)^2$ and $\left(\frac{h_i - \widehat{h_i}}{\widehat{h_i}}\right)^2$ rather than $w^i - \hat{w}^i$ and $h^i - \hat{h}^i$ which helps to reduce the effect of different sizes of an object of the same kind. S^2 Denotes the grid cell B denotes the bounding boxes and 1_i^{obj} denotes the object existence in cell i or not.

D. Transfer Learning

A transfer learning method developed to attain better performance with more transferred feature layers. Transfer learning is being used to extract the features from a custom dataset automatically with the help of using pre-trained models. It is a suitable way to apply transfer learning without considering substantial datasets, training, and calculation, which only consumed the time. Transfer learning is an adequate method if one has a small-scale sample dataset. Transfer learning used pre-trained CNN architecture, where almost 1.2 million samples of ImageNet dataset and 1000 classes have trained with powerful features extraction potential.

TABLE. I. ALGORITHMS PARAMETERS

Algorithms parameters	
YOLOv3	***Improved YOLOv3***
9 anchor boxes	12 anchor boxes
Default loss function	Improved loss functions
Darknet53, 53 Conv layers with 3 YOLO layers	Darknet53, 53 Conv layers, with 4 YOLO layers
Configure Upsampling layers,	Configure Upsampling layers,
Configure residual blocks, Conv 3×3, 1×1, concatenate	Configure residual blocks, Conv 3×3, 1×1, concatenate
Fine-Grained Features	Deep Fine-Grained Features
Batch Normalization	Batch Normalization
CUDA implement in the algorithm	CUDA implement in the algorithm
Train from scratch	Train using transfer learning,fine tuning
No FPS	Compute FPS in the algorithm
Single image size training	Multiscale image training
Train without data augmentation	Train with data Augmentation
Default batch size	Change batch size
No tensorboard Visualization	Tensorboard visualization in code
3 detection scales	4 detection scales
Configuring IOU, mAP	Configuring IOU, mAP

We proposed and trained darknet-53 backbone architecture, which is pre-trained on the ImageNet dataset to extract the features. Then we performed target detection on the COCO dataset by fine-tuning. During the fine-tuning, we adjust several parameters, including the multi-scale size of input images, learning rate, batch size, to boost and enhance the accuracy and performance.

IV. Dataset Composition

The dataset is a key for object detection, and the collection of the dataset is an important, challenging milestone for object recognition. We used four kinds of fish, including anemone-fish, jelly-fish, star-fish, and shark. The samples of the dataset collected from various resources. All the samples of the dataset have varying sizes, such as 320×320, 416×416, and 480×480. The sample of the collected dataset is shown in Table II.

Dataset annotation is a very time consuming process that takes much time than usual. As we know that the fish postures slightly and haphazardly change due to their free and multiple dimensional rotations, so the bounding box labeling inserts with much care and accurate for mAP. Fish move freely, so we need to insert bounding box labeling in each direction for precise detection. We use a labeling tool for dataset annotation, Labelimg.

TABLE. II. The Number of Training and Testing Dataset of Fish Species

Class	Training images	Testing Images	Total Images
Anemone Fish	950	200	1150
Jelly Fish	1005	200	1205
Star Fish	1100	200	1300
Shark	950	200	1150

V. Results and Comparison

The experiment performed by the DL open-source library TensorFlow 1.11, OpenCV 4.1.1, and coding concluded with the high-level language python 3.5 at Ubuntu 18.04 operating system. Training and testing performed on the system intel core i-7-7700, GPU GTX 1080 with 12 GB of memory. Libraries, packages, and hardware specifications are shown in Table III.

We used the MS-COCO dataset for restoring and initialization of darknet-53 backbone architecture for Fish detection tasks. We set the resolution of the image is 608×608 during training the model. At the training stage, the initial and end learning rate set to 1e-4 and 1e-6, respectively, Intersection over Union (IOU) threshold value 0.5, average decay 0.9, and the batch size is 4. We trained our model to 100 epochs. To prevent the model from non-convergence, the learning rate during the training process changed gradually. The hyperparameters showed in Table IV.

In the experiment, we used custom fish detection dataset that consists of 4 classes, such as anemone-fish, star-fish, jelly-fish, and shark. The total number of training images is 4005, and images for testing are 800. The mAP of the proposed model increased, with improved detection scale, k-means++ clustering, loss function, and transfer learning technique of improved YOLOv3, by 4.13% compared to that of baseline YOLOv3, and the detection speed is 39 FPS, which enables real-time detection of YOLOv3. Some state-of-the-art architectures and detectors were choosing for comparisons such as Faster RCNN and YOLOv2 with our improved YOLOv3 model. The mAP with input image sizes of diverse network structures is shown in Table V, and the AP% comparison of YOLOv3 and improved YOLOv3 with our custom dataset and brackish dataset [21] is shown in Table VI.

TABLE. III. Software and Libraries

Tensor flow	1.12
OpenCV	4.1.1
Python	3.6.5
Matplotlib	3.1.2
Numpy	1.16.4
System	Intel Core i7-7700
CPU	3.6 Ghz
GPU	GeForce GTX 1080 Ti Memory 11 GB
CUDA	9.2, 10.0
cuDNN	7.6.0

TABLE. IV. The Hyperparameters

Parameters	Values
Initial learning rate	1e-4
End learning rate	1e-7
Total epochs	100
Warm-up epochs	2
1st stage epochs	50
2nd stage epochs	50
Batch size	4
Image train size	608×608
IOU threshold value	0.5
average decay	0.9995
Gradient optimizer	Adam optimizer
Train mode	GPU

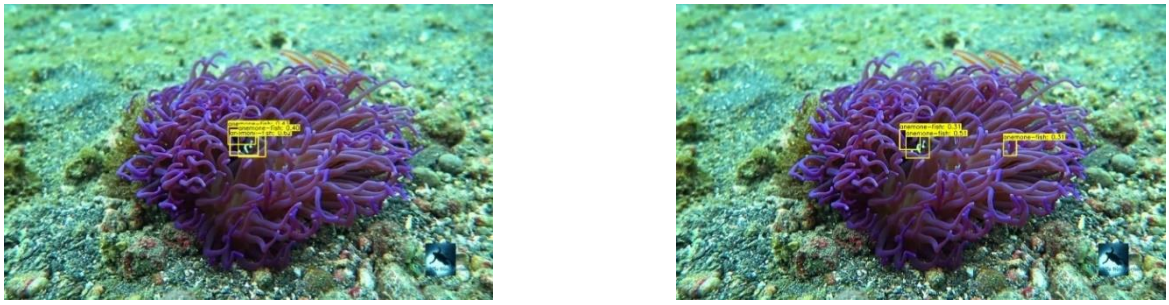
TABLE. V. YOLOv3 Comparison with others object Detector Models

Detection Model	Faster R-CNN	YOLOv2	YOLOv3	YOLOv3 Improved
Input Image Size	480	416	608	608
mAP	77.4%	81.63%	87.17%	91.30%

TABLE. VI. AP (%) of different Fish Dataset Species Comparison between YOLOv3 and Improved YOLOv3

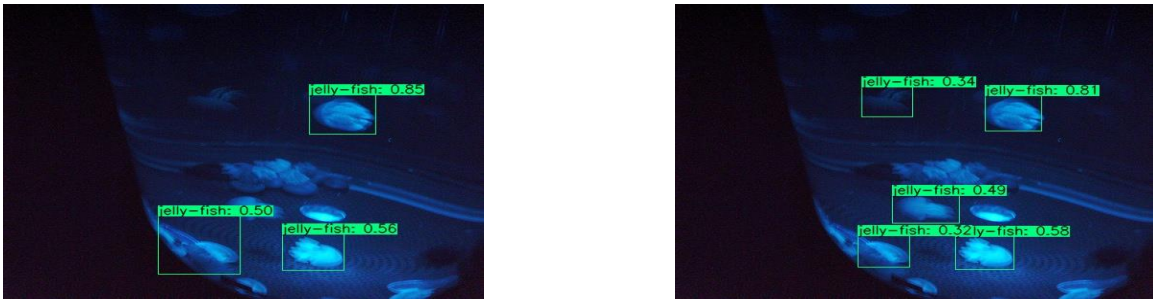
Model	Anemone-fish AP%	Jelly-fish AP%	Star-fish AP%	Shark AP%	mAP
YOLOv3	83.63%	88.21%	88.97%	87.89%	87.17%
YOLOv3 (brackish dataset)	89.99%	82.05%	93.67%		82.17%
YOLOv3 Improved	94.42%	86.14%	98.27%	86.35%	91.30%

The brackish dataset is a publically open dataset that collected from turbid water. Due to its turbidity, small size, the mAP evaluation on this dataset is comparatively less than our custom dataset. The brackish dataset contains 6 classes. We choose 3 classes among them, such as anemone-fish, jelly-fish, and star-fish to check the mAP and AP% on each class at the YOLOv3 detection model. The visualization comparison results between YOLOv3 and improved YOLOv3 are illustrating in Fig. 3, Fig. 4, Fig. 5 and Fig. 6. We draw the curves of model learning loss, confidence loss, and probability loss in Fig. 7, Fig. 8 and Fig. 9. The confidence loss curve of the trained model expresses the object confidence loss at each iteration, which is gradually improving after every iteration. The probability loss curve expresses the probability of an object, either the object belongs to anemone-fish or star-fish. The total loss curve expresses the feature extraction ability of model and model convergence.



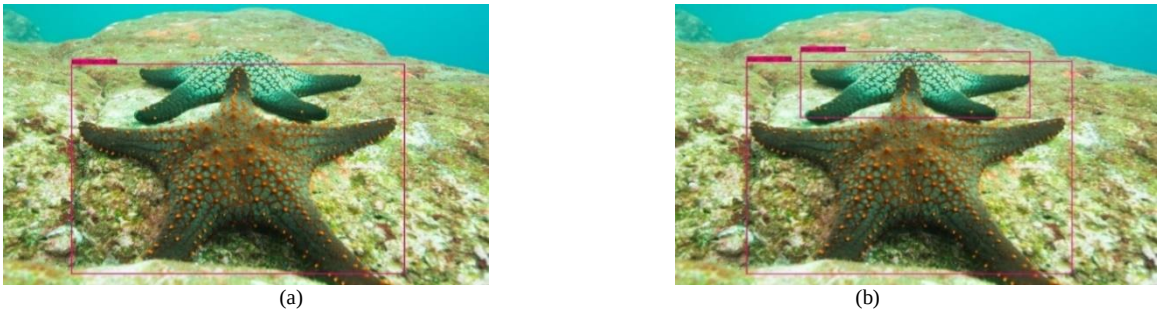
(a) (b)

Fig. 3. (a) Anemone Fish Result of Original YOLOv3, (b) Anemone Fish Result of Improved YOLOv3.



(a) (b)

Fig. 4. (a) Jelly Fish Result of Original YOLOv3, (b) Jelly Fish Result of Improved YOLOv3.



(a) (b)

Fig. 5. (a) Star-Fish Result of Original YOLOv3, (b) Star-Fish Result of Improved YOLOv3.

(a) (b)

Fig. 6. (a) Star-Fish Result of Original YOLOv3, (b) Shark Result of Improved YOLOv3.

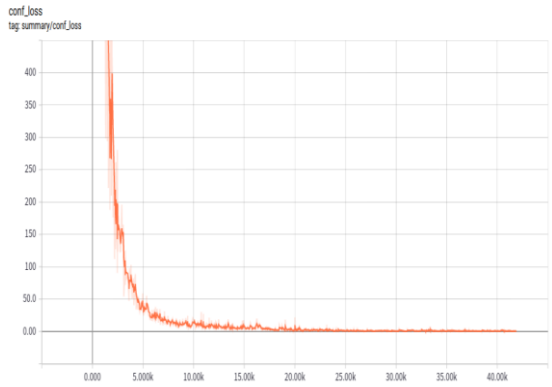


Fig. 7. Confidence Loss Curve of the Trained Model.

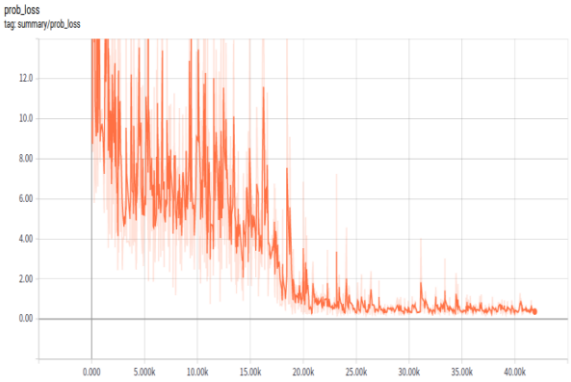


Fig. 8. Probability Loss Curve of the Trained Model.

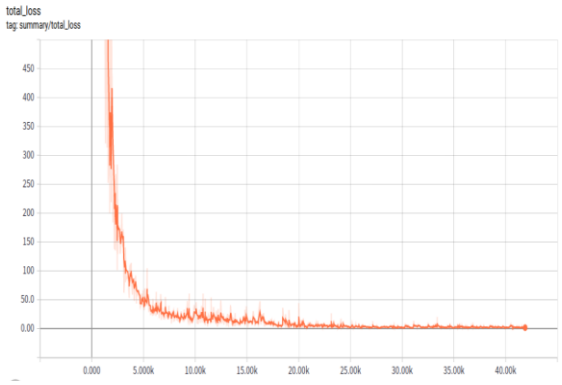


Fig. 9. Total Loss Curve of the Trained Model.

A. Evaluation Matrices

Intersection over union (IOU) and precision, recall are important metrics for model evaluation. IOU is the difference between ground truth and the predicted value which is defined as.

$$IOU = \frac{B \cap C}{B \cup C} \quad (5)$$

Where B is the ground truth value of an object, and C is the predicted value. From the results, it can be clear that the IOU of small and medium size objects is improved by the improved YOLOv3 model than the baseline YOLOv3 model with the addition of a 4^{th} detection scale. Because it has the ability to extract finer grained features of small objects. In summary, the IOU value of the improved YOLOv3 model has greatly improved and better compared to the baseline YOLOv3 detection model. The IOU values of original YOLOv3 and improved YOLOv3 of various objects are shown in Fig. 10.

The precision (P) and recall (R) have been calculated on the basis of true positive (TP) false positvie (FP) and false negative (FN). The precision and recall are defined in equations (6) and (7).

$$P = \frac{TP}{FP+TP} \quad (6)$$

$$R = \frac{TP}{FN+TP} \quad (7)$$

Where TP is the detection of an object correctly with a positive sample, and FP is the detection of an object negatively by the mistake of a positive sample. FN is not detected of an object with a positve sample.

The trade-off between precision-recall is a complicated problem. The precision-recall is one of the significant measures to evaluate the network performance at the testing dataset. In addition, precision is measured with respect to relevancy in results, while recall measures the total number of true, relevant results. The precision-recall curve expresses in the y-axis and x-axis, respectively. The precision-recall curve showed the trade-off at fixed IOU thresholding value 0.5. It is clear from the results with higher precision the recall rate also goes higher, which shows that our model is efficient and converges well. We noticed that the improved YOLOv3 model precision-recall of anemone-fish and star fish is much better than the baseline YOLOv3 model because these two classes have small and extra small objects. In the case of the jelly-fish, the precision-recall curve has not a big difference, and the shark precision-recall curve has been decreased because the size of the object of the shark class is comparatively big than other classes. The precision-recall curves of all fish classes, both YOLOv3 and improved YOLOv3, are shown in Fig. 11.

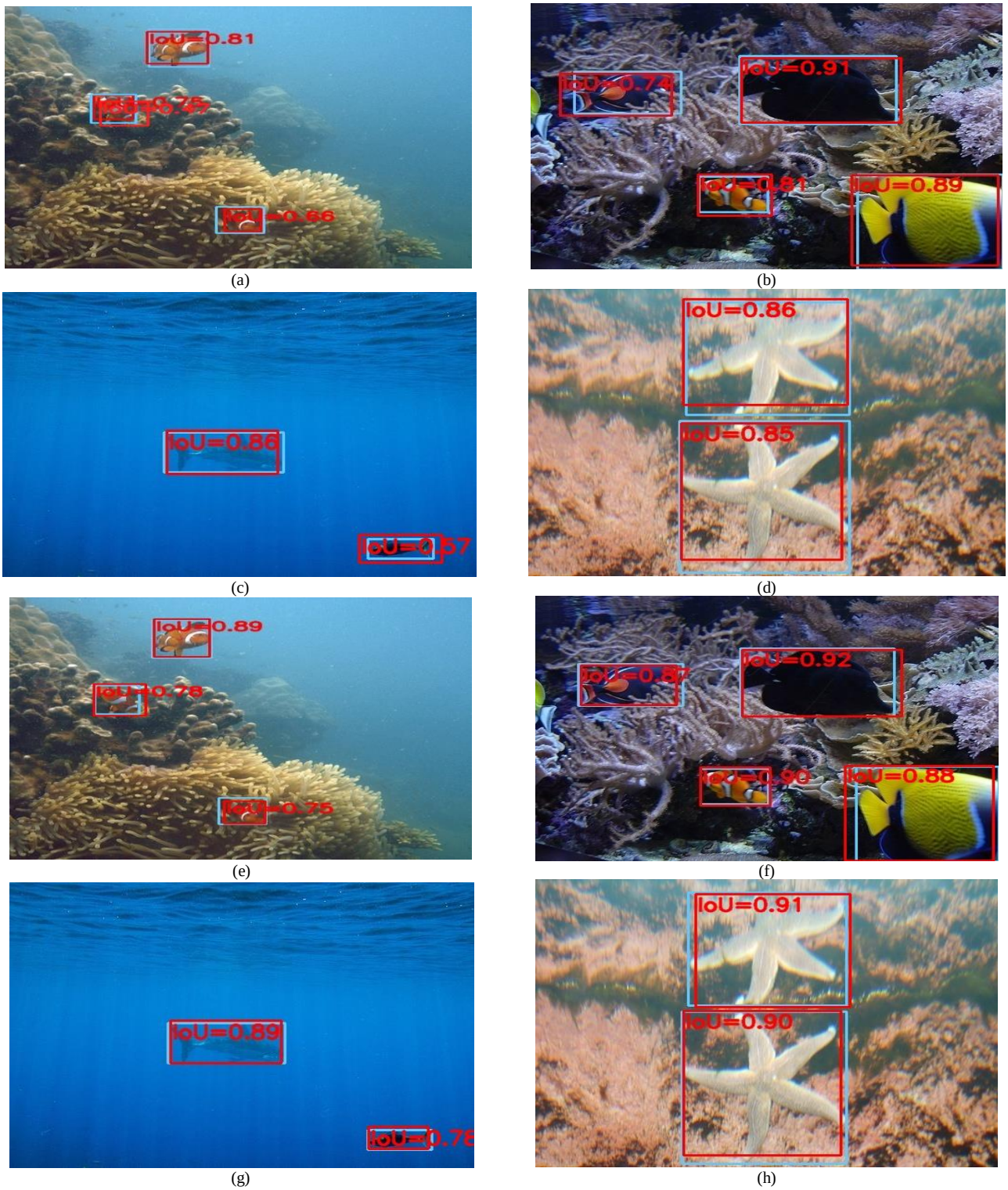
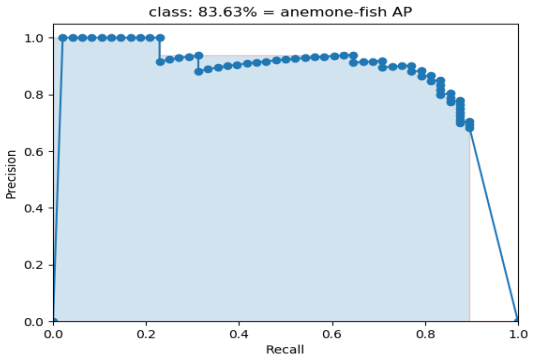
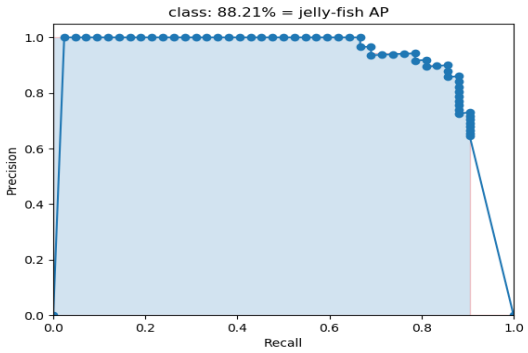


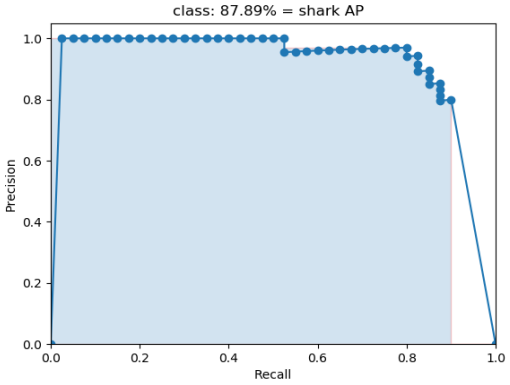
Fig. 10. (a,b,c,d) IOU Values of Original YOLOv3 and in Fig 10. (e,f,g,h) IOU Values of Improved YOLOv3.



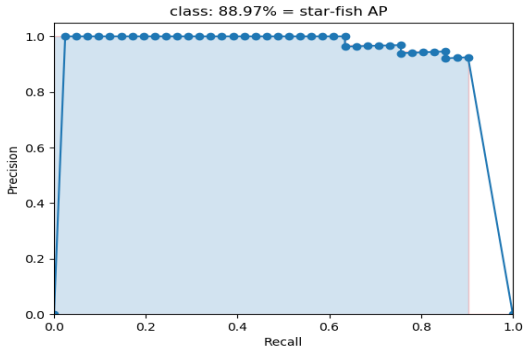
(a)



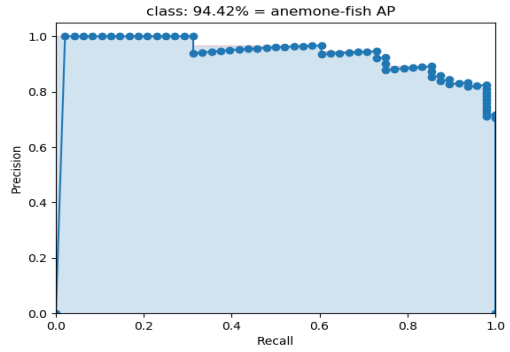
(b)



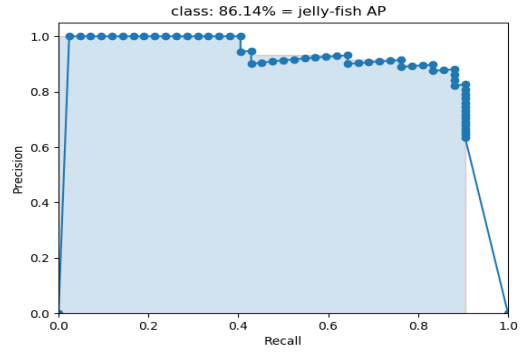
(c)



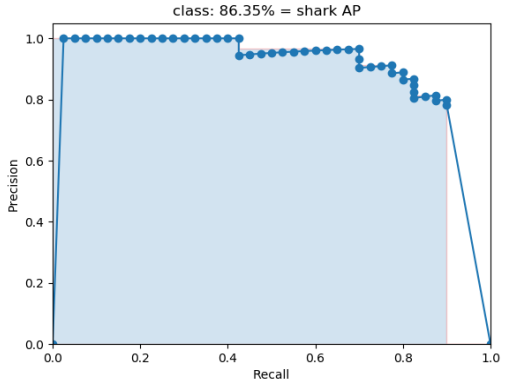
(d)



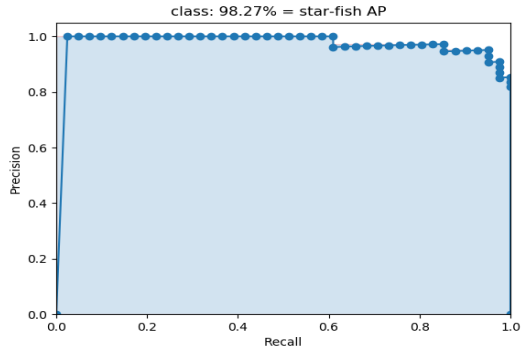
(e)



(f)



(g)



(h)

Fig. 11. (a,b,c,d) AP% Values of Original YOLOv3 and in Fig 11. (e,f,g,h) AP% Values of Improved YOLOv3.

VI. Conclusion and Future Work

Mainly, we introduced how DL could be beneficial for the underwater species analysis at a large-scale dataset. The detection results showed how DL could be achieved excellent results for fish detection. In this paper, we improved YOLOv3 for fish detection. To obtain better results, we increased the detection scale to detect very small size objects. Apply k-means++ clustering to get suitable clusters, as well as transfer learning and improvement in the loss function. The improved YOLOv3 model proves that it outperforms than that of the baseline YOLOv3 model by improving the mAP of 4.13%.

In future work, we will collect large and live datasets, both images and video formats from different underwater conditions. We will improve this model by changing in backbone architecture to make it lightweight architecture and move this model on the embedded system, portable devices for live underwater marine animal detection.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning.” In: Nature 521.7553 (2015), pages 436–444 (cited on pages 21, 126, 128).

[2] Szegedy, C., Toshev, A., & Erhan, D. (2013). Deep neural networks for object detection. In Advances in neural information processing systems (pp. 2553-2561).

[3] Huang, Phoenix X., Bastiaan J. Boom, and Robert B. Fisher. "Hierarchical classification for live fish recognition." In BMVC student workshop paper. 2012.

[4] Bermejo S. (2007). “Fish age classification based on length, weight, sex, and otolith morphological features.”Fish. Res. 84.

[5] R. Larsen, H. Olafsdottir, B.K. Ersbøll, Shape and texture based classification of fish species, Image Anal., 2009, 745-749.

[6] Helge Balk, Development of hydro acoustic methods for fish detection in shallow water, 2001, pg 28.

[7] Fuming Xiang, Application of Deep Learning to Fish Recognition, 2018, pg 53.

[8] River catch of salmon, sea trout and migratory char”, 2019, Available: https://www.ssb.no/en/elvefiske. Accessed on: Dec. 10, 2019.

[9] D.H Hubel and T.N Wiesel. Receptive Fields, Binocular Interaction, and Functional Architecture in the Cat's Visual Cortex. Pages 151-152, 1961.

[10] Adrian Reithaug, Employing Deep Learning for Fish Recognition, 2018, pg 85.

[11] Ekaterina Lantsova, Automatic Recognition of Fish from Video Sequence, 2015, pg 49.

[12] S.O. Ogunlana, O. Olabode , S.A. A. Oluwadare & G. B. Iwasokun, Fish Classification Using Support Vector Machine, 2015, pg 75.

[13] Dhruv Rathi, Sushant Jain, Dr. S. Indu, Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning,

[14] Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

[15] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

[16] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[17] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

[18] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (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).

[19] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

[20] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[21] Pedersen, M. (Creator), Haurum, J. B. (Creator), Gade, R. (Creator), Moeslund, T. B. (Creator), Madsen, N. (Creator) (2019). The Brackish Dataset. Kaggle.