

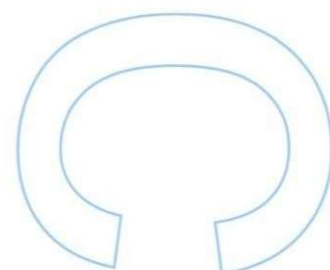
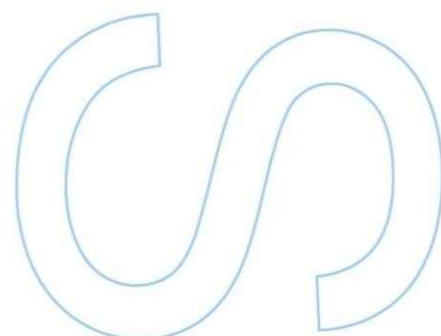
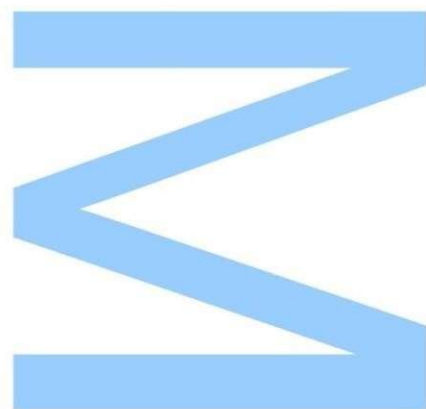
Application of AI Techniques in Video

João Jesus Figueiredo

Master's degree in Mathematical Engineering
Mathematics Department
2023

Supervisor

André Ribeiro da Silva de Almeida Marçal, Assistant Professor, Faculty of
Sciences, University of Porto



U. PORTO
FC FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

M

S

Q

Abstract

Marine animal research plays an important role in human life since the behavior of marine animals is a good indicator of the state of the oceans. However, exploring the underwater environment is a complex task, and the deployment of specialized vehicles guarantees a safe and efficient expedition. These vehicles have several embedded sensors, including cameras that record the environment. As expected, manually analyzing the videos for important events becomes a time-consuming task for researchers, which can be optimized with the application of object detection (OD) models.

A major obstacle arises since these models are highly dependent on high-quality data. Obtaining such data becomes a problem due to the significant impact of the underwater medium on the collected videos. Nevertheless, this study aids researchers in analyzing videos collected from telemetry equipment. Additionally, it is intended to serve as a guide for future studies in underwater marine animal detection using OD models. To achieve this, a dataset is developed consisting of two classes, defined as the Jamanta dataset. Then, the object detection model YOLOv7 is trained with this dataset. In addition, two image processing (IP) methods are applied to the Jamanta dataset, and the effects of training YOLOv7 with the enhanced datasets are discussed.

When training YOLOv7 with the Jamanta dataset, the model has great generalization power, having the possibility to be applied in large-scale marine animal research projects. Comparing the training sessions with and without using the enhanced datasets, the application of IP techniques may even degrade YOLOv7's performance. Such observations invalidate the need for applying these techniques to underwater object detection applications.

Keywords: Underwater Object Detection, Image Processing, YOLOv7

Resumo

O estudo de animais marinhos desempenha um papel importante na vida humana, uma vez que o comportamento animal é um bom indicador do estado dos oceanos. No entanto, a exploração sub-aquática é uma tarefa árdua, e a utilização de veículos especializados garantem uma expedição segura e eficiente. Estes veículos contêm diversos sensores embutidos, incluindo câmaras que gravam o ambiente à volta dos animais. Como esperado, analisar manualmente os vídeos para detetar eventos importantes é uma tarefa demorada para os investigadores, que pode ser otimizada através dos modelos de deteção de objetos (DO).

Os modelos de DO dependem altamente de dados de alta qualidade, que são difíceis de obter num contexto sub-aquático devido ao impacto significativo da água nos vídeos adquiridos. No entanto, este estudo auxilia os investigadores na análise dos vídeos recolhidos por equipamento de telemetria. Além disso, tem o intuito de servir como guia para futuros estudos relacionados com deteção de animais marinhos através de modelos de DO. Para atingir este objetivo, um conjunto de dados é desenvolvido, contendo duas classes e definido como os dados da Jamanta. De seguida, o modelo de deteção de objetos YOLOv7 é treinado com estes dados. Para além disso, dois métodos de processamento de imagem são aplicados aos dados da Jamanta, e os efeitos do treino do modelo YOLOv7 com os dados melhorados são discutidos.

Ao treinar o modelo YOLOv7 com os dados da Jamanta, o modelo apresenta uma grande capacidade de generalização, tendo a possibilidade de ser aplicado em projetos em larga escala, para estudar os animais marinhos. Ao comparar os resultados de treinar o YOLOv7 com e sem os dados melhorados, a aplicação de técnicas de processamento de imagem é capaz de degradar o desempenho do modelo. Tais observações invalidam a necessidade em aplicar estas técnicas num contexto de deteção de objetos sub-aquáticos.

Palavras-chave: Deteção de Objetos Sub-Aquáticos, Processamento de Imagem, YOLOv7

Contents

Abstract	i
Resumo	ii
Contents	iv
List of Tables	v
List of Figures	viii
Acronyms	ix
1 Introduction	1
1.1 Background and Motivation	3
1.2 Thesis' Organization	3
2 Underwater Object Detection	5
2.1 Object Detection	5
2.2 Theoretical Background	7
2.2.1 Convolutional Neural Networks	7
2.2.2 You Only Look Once (YOLO)	11
2.2.3 Challenges and Solutions of Underwater Object Detection	15
3 Materials and Methods	18
3.1 CEiiA's Tagging Equipment	18

3.2	Datasets	19
3.2.1	Jamanta Dataset	19
3.2.2	Aquarium dataset	22
3.3	Underwater Image Processing	25
3.4	Technology used	27
3.4.1	Training the YOLOv7 model	28
4	Results and Discussion	29
4.1	Model Hyperparameter Selection	29
4.2	Training Results and Discussion	37
4.3	Underwater Image Processing	47
5	Conclusions	50
A	Results of Training YOLOv7	53
A.1	Aquarium dataset	53
A.1.1	Test dataset	53
A.2	Jamanta dataset	57
A.2.1	Validation dataset	57
A.2.2	Test set	60
	Bibliography	63

List of Tables

2.1	Equations of MSE, PSNR and Entropy	17
3.1	Distribution of the classes of the Aquarium dataset for the training, validation and test sets	23
3.2	Entropy of the (Red, Green, Blue) channels of the reference and enhanced images, with their respective MSE and PSNR values.	27
4.1	BS, LR, WD, Mom. and IR range values used for the subsequent analysis, where the bold values are the default values from YOLOv7.	29
4.2	Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations	30
4.3	Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations	32
4.4	Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations	35
4.5	Values for BS, LR, WD, Mom. and IR selected	37
4.6	Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations	48

List of Figures

2.1	Generic CNN architecture [46]	8
2.2	Max-pooling and Average-pooling operations [51]	9
2.3	Intersection over Union (IoU) [53], where green represents the prediction made by the model and the red box is the ground truth	11
2.4	The YOLO model [31]	12
2.5	YOLOv7 architecture [37]	13
2.6	Representation of absorption and scattering phenomenons once light travels underwater [70]	16
3.1	Torch tag	19
3.2	Frames from Torch Tag: (a) Frame from Torch tag towed to Jamanta (b) Frame from Torch Tag where a Jamanta is hardly seen.	20
3.3	Example of annotated images	21
3.4	Boxplots with the intensities of each color channel (Red, Green, Blue) for the Jamanta dataset	21
3.5	Boxplots with the width (a) and height (b) of the bounding boxes for the Fish and Jamanta class	22
3.6	Example of images from the Aquarium dataset	23
3.7	Boxplots with the intensities of each color channel (Red, Green, Blue) for the Aquarium dataset	24
3.8	Boxplots with the width (a) and height (b) of the bounding boxes for all the classes of the Aquarium dataset	24
3.9	Enhanced images applying the Xiang et al. [77] (a) and Ghani & Isa [78] (b) methods, applied to image from Figure 3.2 (a)	26

3.10	Boxplots with the intensities of each color channel (Red, Green, Blue) for the Jamanta dataset processed with the Xiang et al. [77] method (a) and the Ghani & Isa [78] method (b)	26
4.1	Plots of mAP@0.5:0.95 values (a) and classification loss for the Aquarium’s validation set (b) for BS = 16 and proposed LR values.	32
4.2	Plot of the objectness loss for the Aquarium’s validation set with LR = $\{10^{-2}, 10^{-3}, 10^{-4}\}$ and BS = 64.	33
4.3	Plots of mAP@0.5:0.95 values (a) and classification loss (b) for the Jamanta’s validation set for BS = 64 and LR = $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$	34
4.4	Plot of the objectness loss for the Jamanta’s validation set with LR = $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ and BS = 64.	34
4.5	Plots of the objectness loss for the Aquarium’s (a) and the Jamanta’s (b) validation sets for WD = 0 and Mom. = $\{0.85, 0.90, 0.937, 0.95, 0.99\}$	36
4.6	Confusion matrix for all the classes of the Aquarium’s validation data set	38
4.7	Plot for the Precision curve measured over increasing confidence thresholds for the Aquarium’s validation data set	39
4.8	Plot for the Recall curve measured over increasing confidence thresholds for the Aquarium’s validation data set	40
4.9	Plot for the PRC curve measured over increasing confidence thresholds for the Aquarium’s validation data set	41
4.10	F1-curve measured over increasing confidence thresholds for the Aquarium’s validation data set	42
4.11	Confusion matrix for all the classes of the Aquarium’s test data set	43
4.12	Confusion matrix for all the classes of the Jamanta’s validation data set	44
4.13	Ground-truth labels (a) and model’s predictions for the Jamanta’s validation data set	45
4.14	Confusion matrix for all the classes of the Jamanta’s test data set	46
4.15	F1-curve for all the classes of the Jamanta’s test data set	47
A.1	Plot for the Precision curve measured over increasing confidence thresholds for the Aquarium’s test data set	53
A.2	Plot for the Recall curve measured over increasing confidence thresholds for the Aquarium’s test data set	54

A.3	Plots for the Precision-Recall (a) and F1-score (b) curves for the Aquarium’s test dataset	55
A.4	Plots for the Precision (a) and Recall (b) Recall curve measured over increasing confidence thresholds for the Jamanta’s validation dataset	57
A.5	Plots for the Precision-Recall (a) and F1-score (b) curves for the Jamanta’s validation dataset	58
A.6	Plots for the Precision (a) and Recall (b) Recall curve measured over increasing confidence thresholds for the Jamanta’s test dataset	60
A.7	Plot for the Precision-Recall curve for the Jamanta’s test dataset of the Jamanta’s test dataset	61
A.8	Ground-truth labels (a) and model’s predictions for the Jamanta’s test dataset .	62

Acronyms

ANN	Artificial Neural Networks	IQM	Image Quality Metrics
AP	Average Precision	IR	Image Resolution
AUV	Autonomous Underwater Vehicle	LR	Learning Rate
AUC	Area under the curve	mAP	Mean Average Precision
BS	Batch Size	Mom.	Momentum
CEiiA	Centre of Engineering and Product Development	msCOCO	Microsoft Common Objects in Context
CNN	Convolutional Neural Networks	MSE	Mean Squared Error
CV	Computer Vision	NN	Neural Networks
DCNN	Deep Convolutional Neural Network	NR	No Reference
DL	Deep Learning	OD	Object Detection
DST	Data Storage Tags	PRC	Precision \times Recall curve
E-ELAN	Extended Efficient Layer Aggregation Networks	PSNR	Peak Signal to Noise Ratio
FFNN	Feed-Forward Neural Networks	ReLU	Rectified Linear Unit
FN	False Negatives	RPN	Region Proposal Network
FP	False Positives	RoI	Regions of Interest
FR	Full Reference	ROV	Remotely Operated Vehicles
FPS	Frames per Second	TP	True Positives
IoU	Intersection over Union	UOD	Underwater Object Detection
IE	Image Enhancement	WD	Weight Decay
IP	Image Processing	YOLO	You Only Look Once

Chapter 1

Introduction

Studying animals is crucial to obtain true data related to how they behave in their natural environments, enabling studies that focus on the animals as individuals, as a population, or even as a community [1]. In underwater exploration, understanding how marine animals interact with each other and the surrounding ecosystems is of utmost importance to human life [2]. Studying the movement, migratory habits, habitat utilization, and other factors of marine animals can provide valuable insights applicable to the entire species. Therefore, there is the need to acquire relevant data to facilitate these analyses.

Before the 1950s, the methodology used with this purpose in mind depended on direct observation methods [3], which faced many challenges because some species are shy and avoid humans, while other species live in habitats that make it nearly impossible for a study to be conducted [4]. To address this problem, telemetry started being used in marine animal research after the 1950s, and it has helped researchers optimize their studies ever since.

Telemetry devices obtain information on free-ranging animals, such as extensive long-term data on marine animals' movements, physiology, and/or environmental parameters, that can be collected from devices called Electronic Tags [5]. Deploying Electronic Tags on ocean exploration missions enables the protection of sub-aquatic ecological environments and the prediction and prevention of underwater disasters, to name a few applications. These devices either transmit the data to a receiver, as is the case with Acoustic and Radio transmitters [3], or log the data, as is the case with Data Storage Tags (DSTs) [5]. Other devices such as Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs) are also currently used for ocean exploration purposes [6], collecting important data from several embedded sensors.

Camera systems may be incorporated into the vehicles, which records the animals' surroundings, allowing researchers to study marine animals in their natural environments and enabling them to understand the social interactions within and between other species [7]. Additionally, by studying directly on the species' habitat, the animals behave as naturally as possible, with little to no human interaction, which is known to bias the data collected from these studies.

However, underwater missions usually last for several hours, meaning that large amounts of

videos need to be analyzed to encounter important events for the research study. As expected, this process is time and staff-consuming since it is conducted almost frame by frame, typically by a researcher. Therefore, there is a need for alternative approaches that guarantee the videos are efficiently and precisely analyzed, which may be achieved by applying computer vision (CV) methods [8]. In this way, researchers may focus on other tasks, while the video is computationally analyzed.

CV is a field of Artificial Intelligence that enables computers and machines to understand the contents and patterns of digital data [9], i.e. images and videos, in an attempt to replicate the analysis based on the human visual system. Traditional CV models relied on handcrafted features, which generalized poorly on unfamiliar data [10]. On the other hand, Deep Learning (DL) methods significantly improved the performance of CV models, using Artificial Neural Networks (ANNs) to reproduce how the human brain works [11]. By applying DL to CV, the model learns the necessary features during implementation, producing more accurate results on unfamiliar data, with no human interaction during the learning phase [12].

Object detection (OD) is a commonly discussed CV field since it deals with several real-world applications, such as autonomous driving [7], text detection [12], face recognition [13], and pedestrian detection [14]. OD combines object localization and classification tasks, where the former accurately locates the object, using a rectangular box, typically defined as a bounding box [15]. The latter consists of assigning a class to an object from a set of pre-defined classes, where a confidence score is associated with each class, and the class with the highest score is the class that the model will predict [10]. Even though OD is known to be one of the hardest CV tasks, significant progress has been made and the field is developing at a fast pace, especially by incorporating DL methods to OD.

Most of the recent OD applications depend on real-time detections, meaning that efficiency is a key factor. However, because of a lack of computation power to process the models [16] and/or due to the need for large amounts of high-quality data [10], such efficiency may not be achievable. Furthermore, once the videos are collected underwater, their quality is highly affected by the underwater medium due to absorption and scattering phenomena. Absorption occurs once light enters the underwater medium, attenuating mostly colors with the largest wavelength (i.e. red and yellow), while colors with the lowest wavelength (i.e. green and blue) are less attenuated [17], explaining why underwater images are typically dominated by green and blue colors. Scattering occurs since the light trajectory changes once it travels underwater, leading to noisy images with a blurred effect and low contrast. By combining both phenomena, low-quality underwater images are obtained, negatively influencing the performance of an OD model [18].

Therefore, high-quality underwater datasets are necessary to improve the underwater exploration field. One way to achieve this is to apply image processing (IP) techniques. These techniques enhance the images, making them more visually appealing, with increased contrast, which is ideal for an underwater image. However, previous studies [6, 19] have noted that applying IP techniques does not guarantee a significant change in the performance of an OD model, having the possibility to reduce its overall performance.

1.1 Background and Motivation

Between October 2022 and July 2023, the Centre of Engineering and Product Development (CEiiA), partnering with Faculty of Sciences of University of Porto, held an internship to apply OD models to detect the animals from videos recorded from Electronic tags developed by CEiiA. This is a project in collaboration with the University of Azores to aid them in their research on the marine animals that inhabit the Azores' islands, i.e. jamantas, tunas, whale sharks, remoras and other animals. CEiiA has developed four types of tags: Torch, Wave, Limpet, and Oyster. These tags have a set of sensors, capable of capturing both recording and environmental data, such as the water's temperature and depth. The tags are first attached to the animal, then the data is collected by the tag until it releases from the animal, lasting up to 8 hours attached to the animal. Then, the tag is retrieved by specialists and the data is analyzed.

The problem arises with the video analysis since the time of the videos corresponds to the duration of the mission, i.e. the time the tag was attached to the animal. By combining multiple missions, the number of videos to analyze increases significantly, which becomes time-consuming for researchers in the long term.

The main contribution of this work lies on developing two datasets from videos collected by the tags: the first dataset consists of images taken from two videos where jamantas and fish are present, both with a duration of over 5 hours. This dataset is defined as the Jamanta dataset. A second dataset is also created taken from several 30-minute videos where whale sharks, fish, and remoras are present. This dataset is defined as the Whale Shark dataset. The frames for the datasets were acquired by manually analyzing the entirety of the videos, where they were selected once an animal was present. Then, the OD model YOLOv7 is trained using this dataset and the results are analyzed. Due to lack of time, the results of training YOLOv7 with the Whale Shark dataset were not acquired, but the dataset is still available to be used in future training sessions.

As a secondary contribution, two IP methods are applied to the Jamanta dataset. Then, YOLOv7 is trained with both enhanced datasets and the results are compared. This study is conducted to verify if the creation of visually appealing images for humans has a significant difference on the performance of an OD model. As a result, the need to apply IP techniques as a pre-processing step is verified and the effects of training YOLOv7 with enhanced datasets are evaluated.

1.2 Thesis' Organization

This thesis is organized as follows: **Chapter 2** covers the topic of Underwater Object Detection (UOD), where OD is explained chronologically. In the same chapter, all the theoretical background necessary for the understanding of the remaining chapters is presented. More specifically, the theory behind Convolutional Neural Networks (CNNs), the You Only Look Once (YOLO) models and the Challenges and Solutions of Underwater Image Processing are presented in **Chapter 2**.

In **Chapter 3**, the methodology is explained, namely the development of the datasets, joined by their analyzes. Additionally, two IP methods are presented and applied to the Jamanta dataset. All the materials are also detailed, such as the tagging equipment and technology used. **Chapter 4** starts by selecting the best hyperparameter combination. Then, the results of the training sessions with these hyperparameters are discussed. In **Chapter 5** the conclusions of this thesis are presented. Finally, **Appendix A** shows the plots and images of training the model with the best combination of hyperparameters, selected in **Chapter 4**.

Chapter 2

Underwater Object Detection

This chapter introduces the most important concepts related to the development of this thesis. OD is first introduced, and is explained chronologically. Then, the theoretical background is presented, where Convolutional Neural Networks (CNNs) and the most important metrics used to evaluate the performance of an OD are explained. Furthermore, the You Only Look Once (YOLO) models are explained in detail. Finally, the challenges of underwater OD are described and a solution for these challenges is presented.

2.1 Object Detection

Object Detection has gone through two different stages since the beginning of the millennium: the traditional object detection period and the deep learning object detection period [12]. Traditional methods depended on handcrafted features, where researchers would design sophisticated feature representations, limited by computing resources at the time [12] and the architecture of these object detectors consisted of four layers: the Input layer, the Region Selector layer, the Feature Extraction layer, and the Classification layer [20].

In the early 2000s, several traditional methods were introduced, but the Viola-Jones object detector, the Histogram of Oriented Gradients (HOG) detector, and the Deformable Parts Model (DPM) are the most honorable mentions because of their high performance. The Viola-Jones [21] was introduced in 2001 as a face detector model, and achieved real-time detection by using sliding windows to look for a face in all image locations [12]. In 2005, the HOG detector [22] was motivated by the problem of detecting pedestrians, and was able to detect objects of different sizes and shapes [12], which is still a problem for modern object detectors. In 2008, the DPM [23] was proposed and achieved higher performance compared with other traditional methods at the time, even winning the PASCAL VOC challenge in 2009 [10]. It followed the philosophy of "divide and conquer", where it would detect an object by dividing it into its parts. For example, to detect a human body, DPM would consider a human to be made of a collection of parts, such as the head, arms, legs, and torso [10].

After 2010, OD hit a plateau because the performance of the traditional object detectors was slow and inaccurate, especially on unfamiliar data sets [10]. However, in 2012, the article [24] introduced a deep learning approach to CV, using Convolutional Neural Networks (CNNs, which will be covered with more detail in section 2.2.1). They defined the state-of-the-art at the time and started a new era for object detection, since deep neural networks can learn robust and high-level feature representations of an image, in a much faster and more accurate way than traditional methods [12]. However, the ideas implemented with these methods were the building blocks for DL methods.

Object detector models are divided into one-stage detectors and two-stage detectors. Two-stage detectors first use Region Proposal Networks (RPNs) to generate Regions of Interest (RoI), followed by classification tasks, at a second stage. R-CNN [25], Fast R-CNN [26] and Faster R-CNN [27] are commonly used two-stage detectors [28]. These models lead to high accuracy rates, with the trade-off of being slow [29]. On the other hand, one-stage detectors try to complete the detection process by treating it as a regression problem, predicting the class and bounding box coordinates in one step [12], leading to faster results with a slight loss in accuracy, compared to two-stage detectors [30]. The You Only Look Once (YOLO) series are great examples of one-stage detectors [31–36].

YOLOv1 [31] was proposed in 2016 and its results were significantly more robust than other state-of-the-art OD models, such as Fast R-CNN. YOLO can be implemented to videos at a frame rate of 45 frames per second (FPS), while Fast R-CNN had more accurate results at only 0.5 FPS. This fact enables YOLO to be implemented on real-time applications, such as autonomous driving. However, this model struggled with smaller objects and YOLOv2 [32] was proposed with the objective of fixing the issues from YOLOv1. This version introduced anchor boxes, which are a set of several standard bounding boxes, selected after analyzing the dataset and underlying objects [32]. YOLOv3 [33] and YOLOv4 [34] applied data augmentation techniques, where new images are added to the dataset by applying image transformation techniques to the already existing images. YOLOv5 introduced the anchor box selection process, where the most appropriate anchor boxes are automatically learned by the model [10]. YOLOv6 [35] followed an anchor-free philosophy, but it was considered unstable compared to YOLOv5. YOLOv7 [36] used the ideas of YOLOv4 and YOLOv5, achieving a good balance between detection efficiency and accuracy [37].

These models may also be applied on underwater scenarios. The survey [38] summarized DL approaches for marine object detection. More specifically, it reviewed articles that focused on detecting fish, plankton and corals using one of the models from the R-CNN or YOLO family, with different training strategies. For example, in [39] the authors proposed the *zooplanktoNet*, a DL model to detect zooplankton based on CNNs, where the authors verified that most underwater data sets suffer from class imbalance problems. To fix this issue, the authors applied data augmentation techniques to the data sets. Additionally, the authors from [40] proposed a CNN approach to plankton classification that dealt with class imbalance data sets using transfer learning, a technique used to adapt the parameters learned from previous training sessions to

other similar domains [41].

It is worth mentioning that DL applied to OD depends entirely on massive amounts of training data to guarantee that OD model performs well on unknown data. Considering this requirement, object detectors tend to perform poorly on smaller-sized data sets [14], where there are fewer annotated instances. Almost all data sets used for CV tasks, for example PASCAL VOC [42], ImageNet [43] and msCOCO [44], contain thousands of images with even more annotated objects (since one image can contain more than one object present). The development of these data sets is usually time and staff-consuming, which is a major problem once underwater images or videos are used to train the models.

2.2 Theoretical Background

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of Feed-Forward Neural Networks (FFNNs), composed of neurons that are connected through weights (or learnable parameters), which communicate with each other by receiving weighted information (either from other neurons or external inputs), producing an output that may or may not be used throughout the rest of the network. These networks have the capability of exploring pixel values with high detail, finding abstract features from images, learning from them and obtaining a model that can be used on data unknown by the model. Each feature of an image is associated with a weight. The higher the importance of a feature for the detection, the higher the weight value on the NN.

NNs are commonly organized into stacked layers where the first layer is the input, the last layer is the output, and the layers in between are the hidden layers, where the network acquires all the necessary features for the task at hand [28]. The combination of the input layer, hidden layers, and output layer defines the architecture of a Neural Network. A generic architecture of a CNN is shown in Figure 2.1.

The connections between neurons are unidirectional from inputs to outputs, where the weights are randomly initialized on the first layer, serving as inputs to subsequent neurons, until reaching the last layer, where the outputs of the NN are calculated. Then, the weights are used to compute a loss function's value, which the FFNN has the purpose to minimize. After this step, a backpropagation algorithm is applied, where the weights are updated throughout the network. By repeating these steps for a pre-specified number of iterations (or epochs), the FFNN is training and, hopefully, the performance of the model improves at each step of the process [45].

The first layer of Figure 2.1 is the input layer, where the model initially processes the image. Then, the following layers (the hidden layers) alternate between convolutional layers, pooling layers and other normalization layers, used for feature extraction. The final layers are usually fully connected, that are used for classification purposes [13, 47]. In each convolutional layer,

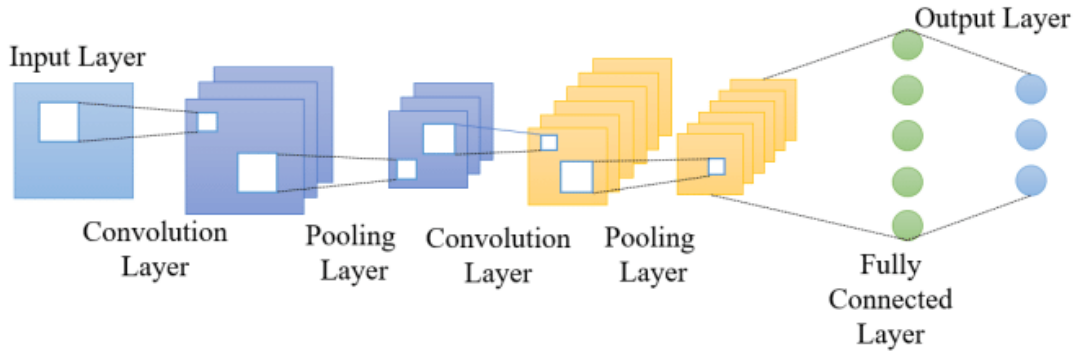


Figure 2.1: Generic CNN architecture [46]

CNNs apply the convolution operation with an activation function, between a set of filters (or convolutional kernels) and a given input, generating an output feature map [48].

The convolution operation is given by

$$X_j^l = \mathbf{f} \left(\sum_{i \in M_j} X_i^{l-1} * W_{ij}^l + b_j^l \right) \quad (2.1)$$

where W^l is the weight matrix of the feature map corresponding to the l th layer, $*$ is the convolution symbol, a bias b_j^l is added to the results of the convolution operation and $f(x)$ is a nonlinear activation function.

The activation functions are commonly used in CNNs since they first produce a linear output and these functions convert them into non-linear outputs, enabling the network to capture non-linear relationships between the input features, learn better patterns from the data, and decide whether a neuron should be used or not [49].

The Rectified Linear Unit activation function was proposed in 2010 [50] and is the most commonly used activation function, since it offers better performance and generalization compared to other activation functions, such as the Sigmoid and Hyperbolic Tangent functions. The ReLU activation function is given by

$$f(x) = \max(0, x) \quad (2.2)$$

However, when the ReLU is used as the activation function, some neurons fail to activate since their gradient values are set to zero at earlier stages of training. To fix this issue, the Leaky ReLU activation function was proposed in 2013 [49], which introduced the parameter α to guarantee that the gradient values are never set to zero. This function is given by Equation (2.3)

$$f(x) = \alpha x + x = \begin{cases} x, & \text{if } x > 0 \\ \alpha x, & \text{if } x \leq 0 \end{cases} \quad (2.3)$$

After the convolutional layer is applied and the feature map is activated, a pooling layer is used to down-sample the feature maps, reducing their sizes, where a fixed rule is applied. This rule depends on the application at hand, but CNNs usually use the max-pooling operation, where the maximum value of a $k \times k$ neighborhood is applied through all the image, gathering the most relevant features and excluding the rest [28]. The average pooling is another pooling operation, which averages all the values of a $k \times k$ neighborhood, as shown by Figure 2.2 [51]. Convolutional and pooling layers may be combined in several ways. However, once the network reaches the classifier head, the last of the max pooled feature maps in the backbone is flattened. The flattening operation is done by transferring each pixel value of this final feature map to an input neuron with one dimension, serving as input to the fully connected layers [28].

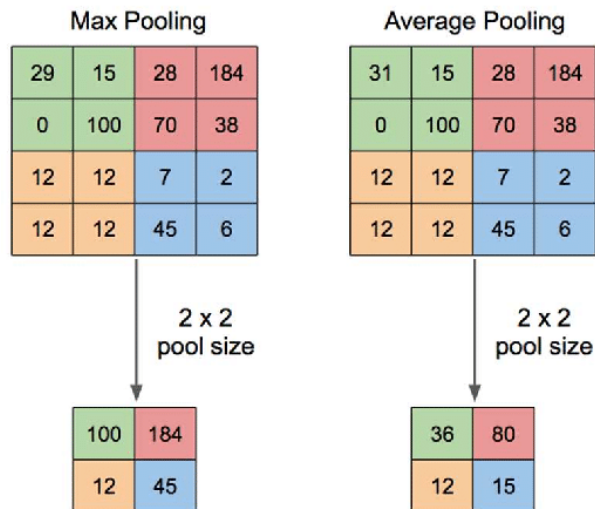


Figure 2.2: Max-pooling and Average-pooling operations [51]

Finally, in the last layer of the fully connected layers, the softmax function is used to compute the probability distribution from x , a vector of real numbers, producing an output ranging between 0 and 1, indicating the probabilities of each class, that must be predefined. The softmax function is given by

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (2.4)$$

where x_i is the i th element of x .

At this stage, the object detection task is concluded and each class has its probability associated. The class with the highest score indicates the one chosen by the model to classify the

object. Additionally, several metrics may be used to verify if the results are appropriate to be used on unseen data and to assess if any problems occurred during the network training stage.

Object Detection metrics

The Mean Average Precision (mAP) is the most common metric used to verify the performance of object detectors, and it is based on the Precision and Recall values, defined by Equations (2.5) and (2.6), respectively.

$$Precision = \frac{TP}{TP + FP} = p \quad (2.5)$$

$$Recall = \frac{TP}{TP + FN} = r \quad (2.6)$$

where TP are True Positives, which are the correct detections made by the model, FP are False Positives, which are objects that were incorrectly identified by the model. The FN are the False Negatives, which are objects that the model did not consider for the detection task [52]. Precision is the percentage of correct positive predictions, measuring the ability of a model to identify only relevant objects. Recall is the percentage of correct predictions among all given ground truths [53]. Both precision and recall are dependent on the confidence associated with each bounding box, generated by the model. These confidence scores are defined as the model's confidence that an object is inside the bounding box [10]. Consequently, a higher confidence score indicates a better localization performance by the model.

Therefore, an OD model is considered a good model if it can find all the ground-truth objects and identify all the relevant objects of the image [54]. This implies that both precision and recall remain high, even if the confidence threshold decreases. By combining both metrics, the Precision \times Recall curve (PRC) is obtained, where a large area under the curve (AUC) indicates both high precision and high recall. From the AUC, it is possible to obtain the average precision (AP). Since the AUC of the PRC is not monotonic, it is interpolated to guarantee the curve's monotony and an accurate AUC. To do this, the maximum precision values are averaged in a set of equally spaced recall levels. Then, the AP is calculated for each class of the dataset. Finally, the mean average precision (mAP) is the average of the AP for all the classes and is given by Equation (2.7), where AP_i is the AP for each of the dataset's class [53].

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (2.7)$$

The Intersection over Union (IoU) is another commonly used metric in OD that measures the accuracy of the identification of the object, by comparing the predicted bounding box with the ground-truth bounding box [7]. With a given threshold t , it may also be used to define

which detections are classified as correct, if $\text{IoU} \geq t$, and incorrect, if $\text{IoU} < t$. Figure 2.3 defines graphically the IoU.

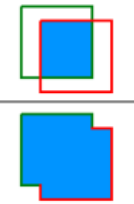
$$\text{IoU} = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


Figure 2.3: Intersection over Union (IoU) [53], where green represents the prediction made by the model and the red box is the ground truth

Furthermore, the Precision and Recall values are also used to define another metric called the F1-score, given by

$$\text{F1-score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2.8)$$

The F1 metric is the harmonic mean between Precision and Recall, meaning that it considers both false positive and false negative values [55]. It is especially used to assess the detection performance over imbalanced data sets, since each class may be individually studied with this metric.

2.2.2 You Only Look Once (YOLO)

CNNs can be further defined as having specific components that deal with specific tasks. Such components are mainly the Backbone, the Neck and the Head [6]. The backbone is one of the most important components of a CNN, which extracts the features from the input image [28], and is commonly represented by Deep Convolutional Neural Networks (DCNNs), such as AlexNet [24], GoogLeNet [56], VGG-16, and VGG-19 [10, 57]. The neck is placed between the backbone and the head and is used to aggregate features collected from feature maps in different stages of the network [14], allowing low-level features to interact directly with high-level features, by mixing their information [58]. Finally, the head, also known as the classifier head, is usually a sequence of fully connected layers, that uses the features extracted from the backbone. Combined with the neck, the head classifies those features into output classes [28]. There are two types of classifier heads: one-stage heads and two-stage heads [12], which were briefly explained in Section 2.1.

One-stage detectors have simpler architectures, enabling them to output bounding boxes and class probabilities, by considering the whole image while visualizing it once [14]. This leads to higher detection speeds, although the detection performance may suffer, compared to two-stage detectors. The YOLO series are great examples of one-stage detectors that started in

2016, outperforming other state-of-the-art object detectors [14], in terms of speed and detection accuracy.

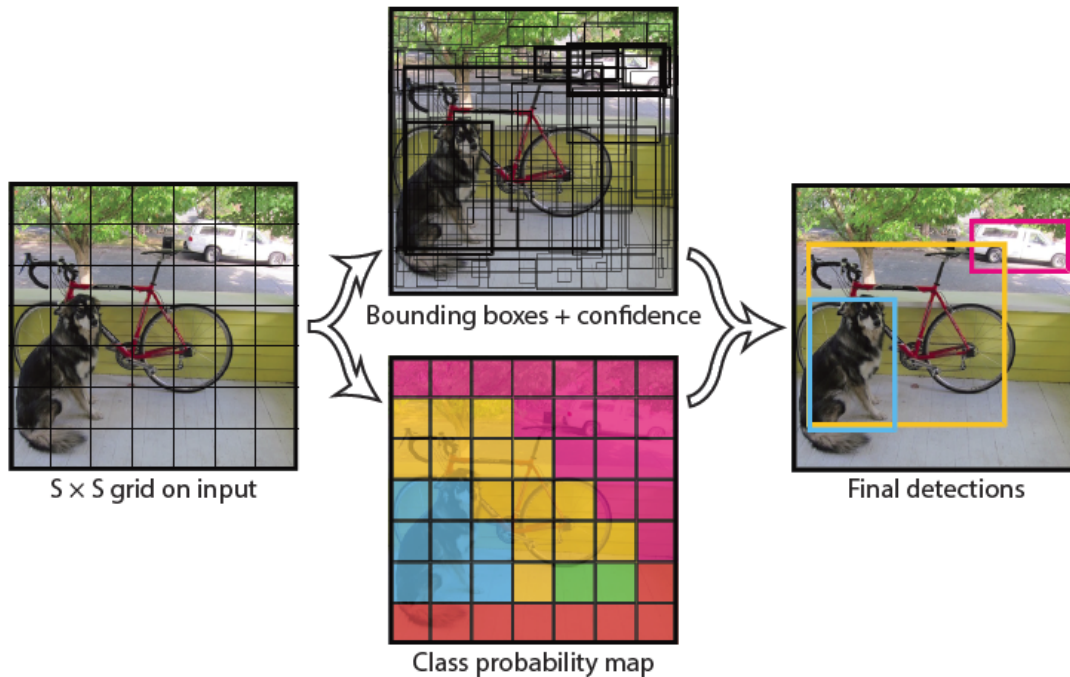


Figure 2.4: The YOLO model [31]

YOLO starts by dividing the input image into an $S \times S$ grid, and each cell is responsible for the detection of an object only if the object's center is located inside the cell. Each grid cell predicts B bounding boxes and confidence scores for each box [31]. The latter is formally defined as $Pr(Object) * IOU_{pred}^{truth}$. $Pr(Object)$ is the probability that the grid contains an object and IOU_{pred}^{truth} is the IoU (Figure 2.3) between the ground truth and the predicted box [31]. The grid cells also compute the conditional class probabilities for each class, $p(c_i)$, where $i = 1, \dots, n$, which is used to verify both the class appearing in the box and how well the predicted box fits the object [31]. Finally, each bounding box predicts a set of parameters: (x, y) , the coordinates of the center of the predicted bounding box; height (h) and width (w), the predicted dimensions of the bounding box; c_s , the confidence, which represents the probability that the grid contains an object. Figure 2.4 shows how YOLO processes an image.

Therefore, YOLO computes $S \times S \times (B \times 5 + n)$ values in total during training. There are $S \times S \times B$ bounding boxes, each bounding box computes five parameters, for the number n of classes to be predicted. This leads to a large number of boxes that need to be filtered so that only the most appropriate bounding boxes are shown in the image. To filter the bounding boxes, the first way is to discard the boxes that have a value of $p(c_i)$ smaller than a predefined threshold, usually 0.5. Then, the non-max suppression algorithm (NMS) is applied to filter the rest of the boxes [14]. More specifically, the bounding boxes generated by the model are independent for each detection, which means that an overlap of detections is bound to occur at least once, as shown in the top part of Figure 2.4. Therefore, once detections are highly overlapped, NMS collapses

them into one detection, significantly reducing the number of detections initially proposed by the model [59].

Several YOLO versions have been developed since 2016. However, this work focuses on implementing YOLOv7 which, during the early stages of the internship, was the latest YOLO version. Ever since, other versions were proposed, such as YOLO-NAS¹ and YOLOv8².

YOLOv7 [36] is a real-time object detector that was trained only on the msCOCO dataset [44]. without using any other data sets or pre-trained weights. The original architecture is presented in Figure 2.5.

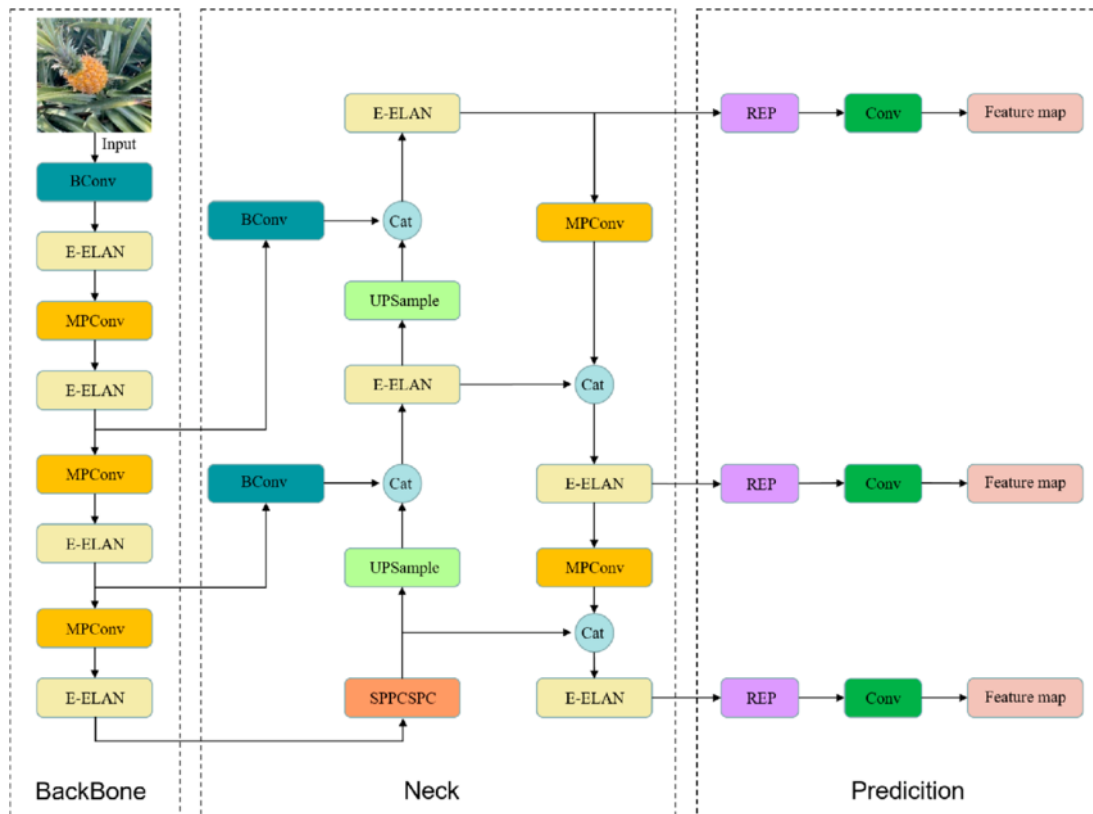


Figure 2.5: YOLOv7 architecture [37]

YOLOv7 follows the basic CNN architecture (Figure 2.1). However, certain changes were implemented to improve the performance of the model, in addition to the changes from previous YOLO models. The backbone is composed of Extended Efficient Layer Aggregation Networks (E-ELAN) modules to complete the main feature extraction task [60], which was proposed to design an efficient network [36]. BConv modules, composed of Convolutional layers, Batch Normalization layers, and LeakyReLU activation function (Equation (2.3)), are included to extract image features of different scales. Batch Normalization is a regularization technique [61] introduced in YOLOv2, which addresses the issue caused by the random initialization of the first layer [14].

¹<https://github.com/Deci-AI/super-gradients>

²<https://github.com/ultralytics/ultralytics>

The Neck is composed of a path aggregation feature pyramid network structure, realizing the efficient integration of features at different levels of the network [37]. Through three different detection heads, the network outputs the feature maps with three different sizes, and these become in the form of a one-dimensional vector from which the classes may be predicted [60].

YOLOv7's loss functions consists of Objectness Loss (L_{obj}), Classification Loss (L_{cls}), and Box/Localization Loss (L_{box}). Objectness loss measures the probability that an object exists in a proposed region of interest. The box loss measures how well the model locates the center of an object and how well the bounding box covers the object. Finally, the classification loss refers to how well the algorithm can predict the correct class of an object [62]. The total loss is given by the sum of these three components [60] as shown in Equation 2.9,

$$Loss = a \times L_{obj} + b \times L_{cls} + c \times L_{box} \quad (2.9)$$

where a, b, c are the weights corresponding to the three partial losses [60].

OD models contain a set of hyperparameters that should be combined to achieve the best possible result. This approach is based on hyperparameter optimization. YOLOv7's optimization algorithm is based on the integrated genetic algorithm [63]. Genetic algorithms are inspired by biological theories that integrate the idea of maintaining the hyperparameters that are more likely to improve the model in future generations [64]. Even though the most appropriate set of hyperparameters is obtained, other alternatives can be explored, since YOLOv7's optimization algorithm takes a large amount of time to complete. Grid search is an example of a more traditional alternative to obtaining the best hyperparameter combination. This technique conducts an exhaustive search, evaluating all the hyperparameter combinations from a grid of configurations [64].

YOLOv7's hyperparameters mainly consist of the Batch Size (BS), Learning Rate (LR), Weight Decay (WD), Momentum (Mom.) and Image Resolution (IR). BS represents the number of images that are processed together during model training [65]. The higher the BS, the more images will be processed at a time, significantly decreasing training time. However, as BS increases, the need for computational memory also increases. The LR determines how much the weights of the neural network change during training [66]. High LR values imply sudden changes once the weights are updated, therefore training becomes unstable. Low LR values greatly influence training time with no certainty that the model converges. WD is a regularization technique that is introduced in the loss function to prevent overfitting and underfitting issues due to lack of data [67]. Furthermore, since the objective of training a neural network is to minimize the loss function, there are several algorithms that deal directly with this optimization problem, such as the Stochastic Gradient Descent (SGD). Having said that, this algorithm has the recurrent problem of assuming that a local minimum is the global minimum, which is not the optimum solution. However, when Mom. is introduced to the SGD algorithm, this problem is mitigated, significantly increasing the model's rate of convergence and accuracy [68]. Finally, IR is associated to the spatial resolution of the images and should also be assessed.

2.2.3 Challenges and Solutions of Underwater Object Detection

When the underwater medium is explored using specialized divers, it usually becomes expensive, considering the equipment and personnel needed to complete those missions. Additionally, the missions become inefficient, given that only a limited amount of time can be spent underwater, leading to an increase in the duration of the research study [69].

Specially designed vehicles are also an alternative such as remotely operated vehicles (ROV) and autonomous underwater vehicle (AUVs) [6]. The deployment of these vehicles on underwater missions enables a further understanding of the underwater medium, since they carry cameras that capture the surrounding environment. However, they also deal with their specific issues, such as hardware limitations, namely depth restrictions and a limited field of view, that influence the quality of the acquired videos and images [69].

Adding to the vehicles' limitations, the quality of the videos collected by the cameras are highly affected by how light travels underwater. As soon as light enters underwater, it is absorbed and scattered, attenuating different colors at different rates [70]. Light absorption is best understood from Figure 2.6 [71], where the colors with the highest wavelengths (red and yellow) are absorbed in lower depths. The colors with the shortest wavelengths (blue and green) are only absorbed in higher depths. This implies that videos and images taken underwater will be biased since blue and/or green colors will be abundant (depending on the types of water the recording is conducted).

As light travels underwater, a scattering effect also occurs, represented by Figure 2.6, where it can be divided into three parts, as described by the Jaffe-McGlamery underwater imaging model [72]: direct scattering of light from the target to the imagery system; forward scattering, which is a small angular deviation of the light affected by the particles in the water, deviating from the path of the direct scattering; backscattering, which occurs due to the existence of underwater particles that light encounters and then scatters.

Therefore, by combining the absorption and scattering phenomena, the quality of the acquired images suffers significantly. For example, the images suffer from color cast, where a color dominates the entire image. Additionally, usually the images appear to be blurred and with low contrast, significantly reducing the amount of information present. These issues lead to low-quality underwater images, which have a negative impact on the performance of an object detection model [18], since they depend on large amounts of high-quality data.

Underwater Image Processing

To make the images more visually appealing, image processing (IP) techniques are usually an inexpensive alternative used to fix the issues of underwater imaging. IP techniques are divided into image enhancement (IE) and image restoration methods. The latter uses physical models which depend on additional information such as the water's turbidity, attenuation coefficients,

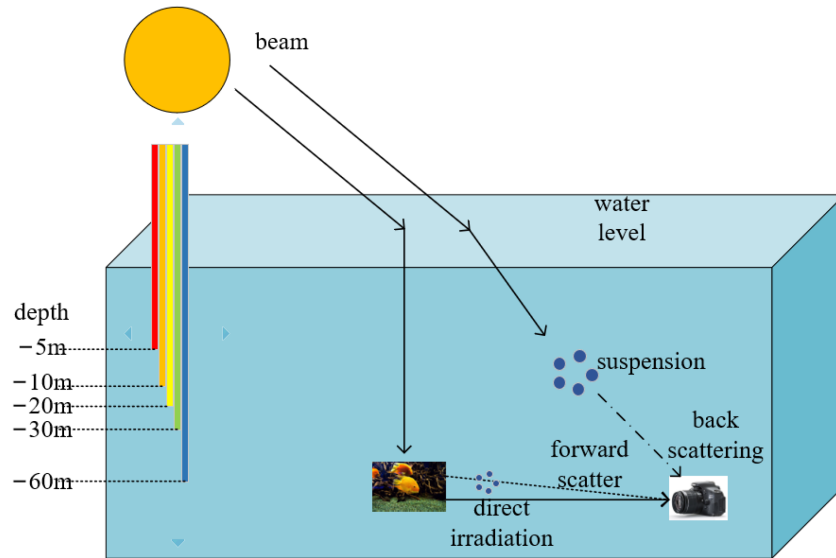


Figure 2.6: Representation of absorption and scattering phenomena once light travels underwater [70]

time of day, and location [73]. The former uses qualitative and subjective criteria to enhance the pixels to produce a pleasant image [70]. These are known to be less computationally expensive than image restoration techniques [73].

By pre-processing the data using IP techniques, it is then possible to train OD models with the enhanced data. Since the images become more visually appealing, with increased contrast and color, it is expected that the results of this experiment improve the performance of the model. However, the studies [6, 19] noted that applying these techniques does not imply such improvement. Having said that, both studies empirically analyzed the effects of IP techniques on model performance, where the authors used two underwater datasets and several IP techniques. Both studies concluded that there is no linear relationship between enhancing the images and the performance of an OD model. Nevertheless, these analyses are empirical, meaning that it is not certain that, in other domains, the application of IP techniques has a positive effect on the performance of an OD model. Therefore, understanding the effect of these techniques on OD models is crucial to achieving the most appropriate results, even though their application comes with great computational costs as the dataset increases.

After applying IP techniques, the enhanced images are usually compared and analyzed with the original, to understand the effects of these techniques. More specifically, qualitative and quantitative analyzes are often carried out. The qualitative analysis consists of visually analyzing the effects of these techniques on the original images, which leads to the choice of the most appropriate technique. However, these analyzes may lead to a bias in the decision of the best method, since opinions vary from person to person. Consequently, quantitative analyzes are also conducted, where image quality metrics (IQM) are used to evaluate the effects of the enhancement. The authors in the article [69] reviewed several IQMs used to evaluate the effects of underwater image enhancement methods. They divide these metrics into two groups: Full Reference (FR)

and No Reference (NR). FR metrics compare the enhanced image with the original image. NR metrics do not need the original image to assess how the IP method performed on the image.

Concerning the work developed within this thesis, Table 2.1 shows only the MSE, the PSNR, and the Entropy metrics. The article [69] presents other important metrics, which are not considered since it is not the focus of this thesis.

Table 2.1: Equations of MSE, PSNR and Entropy

IQM	Definition
MSE	$\frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N [F(i, j) - E(i, j)]^2$
PSNR	$20 \log_{10} \left(\frac{MAX_F}{\sqrt{MSE}} \right)$
Entropy (H)	$-\sum_{i=0}^{255} p_i \log_2 p_i$

MSE is a metric between the reference and enhanced image, which compares the cumulative squared error between both images. In Table 2.1, F is the original image, E is the enhanced image and $M \times N$ is the size of the image. Low MSE means that the enhancement's quality is better [69]. Furthermore, PSNR is the ratio between the pixels' highest value and the noise of the image, where MAX_F represents the maximum pixel value in the equation of Table 2.1. Finally, the entropy (H) measures the amount of information present in the image the higher its value. In the equation of Table 2.1, p_i is the probability of a pixel having intensity i in the image F [69].

Chapter 3

Materials and Methods

In this chapter, the materials and methods used throughout this thesis are presented. Firstly, CEiiA's tagging equipment is presented, where the components of the telemetry equipment are explained. Then, the datasets used to train YOLOv7 are explained in detail, joined by the dataset analysis. This analysis helped solidify the quality of the developed datasets. Then, two IP methods are presented, as an attempt to verify the effects of enhancing the images from the Jamanta dataset on the performance of the model. Finally, the technology used is introduced, joined by a detailed explanation of how YOLOv7 is trained from scratch.

3.1 CEiiA's Tagging Equipment

CEiiA partnered with the University of Azores to aid them in their research on the marine animals that inhabit the Azores islands, specifically jamantas, whale sharks, remoras and other marine animals. To study these animals, four types of electronic tags were developed. These tags are externally attached by specialized divers, in a way that the animals are minimally harassed. This represents a significant progress since it is well known in the marine animal's research community that telemetry techniques can be extremely invasive to marine animals, especially externally attached tags [5]. Consequently, telemetry studies must be clearly defined from the beginning of the research study [74].

Torch and Wave are towed tags, attached around the animal that release after a period of 8 hours, reaching the surface of the ocean and emitting a radio signal so that it can be recaptured. Limpet and Oyster are mounted tags, with suction caps to be mounted on the animal without the need to be around it. However, since these tags release from the animal after a period of 8 hours, they are not suitable for studies where the data is needed to be collected throughout a longer period of time. Additionally, suction tags have been noted to be constrained to 5 hours of use, limiting the amount of data that is collected by these devices [75]. Therefore, other types of tags should be used if the research study needs access to long-term data.

Nevertheless, the focus of this section is on towed tags, since the data collected for the

development of this study came from a towed tag, specifically Torch, presented in Figure 3.1.

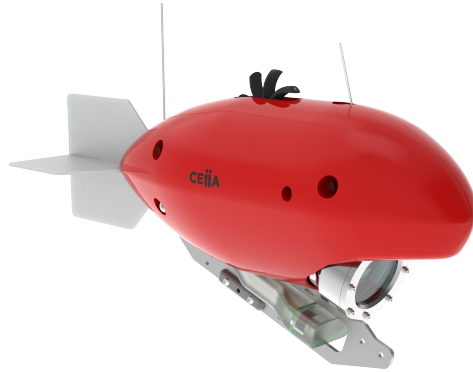


Figure 3.1: Torch tag

Torch weighs 1600g, with $375 \times 120 \times 155$ mm in length \times width \times height, reaching up to 2000 m in depth. This tag embeds environmental sensors, motion sensors and a camera to record the marine animal's environment. The environmental sensors include pressure, which is used to measure depth values, temperature and luminosity (ambient light). The motion sensors include a velocity sensor and Inertial Measurement Unit (IMU), composed of Accelerometer, Magnetometer and Gyroscope, to gather information about the orientation of the tag, which can be inferred to acquire the animal's orientation and its positions [76].

The camera is a Sony IMX219, with an image resolution of 1920×1080 , and video recording at 30 FPS at a distortion level below 1%. This level of distortion is appropriate for recording underwater videos because there are several particles that a camera with higher distortion would focus, instead of mainly focusing on the animal that appears in range of the camera lens.

3.2 Datasets

OD depends on large amounts of data for the model to perform well. Considering the purpose of this work is to detect marine animals on a specific context, a dataset was developed from the videos collected by the telemetry equipment.

3.2.1 Jamanta Dataset

For the Jamanta dataset, the tag is towed to a jamanta. The images were taken from two videos having a duration of over 6 hours, with an image spatial resolution of 1640×922 and a file size of 24.3 GB and 10.1 GB, respectively. This dataset consists of two classes: Fish (0) and Jamanta (1). From the videos, tunas and remoras are present, and these fish were assigned to one class only due to their similarities, while the jamantas were assigned to its respective class.

The FFmpeg¹ platform was used to acquire the frames, which were collected at a frame rate of 3 FPS, 5 FPS and 10 FPS, since consecutive frames do not offer a significant difference in the image. Figure 3.2 shows frames taken from the videos. Figure 3.2 (a) presents a gathering of jamantas, which is rarely captured by the camera, even though jamantas usually travel in groups. Figure 3.2 shows that the quality of the image is affected by the water medium, since blue is the predominant color. This issue leads to a lack of variety in the collected images that are sometimes difficult, even for a human, to correctly identify any animal. Figure 3.2 (b) shows an example of an image where a Jamanta is hardly seen.



Figure 3.2: Frames from Torch Tag: (a) Frame from Torch tag towed to Jamanta (b) Frame from Torch Tag where a Jamanta is hardly seen.

Therefore, the frame acquisition step was difficult to complete, even when the fish or jamanta were in front of the camera. The videos include a legend on top, as shown in the Figure 3.2 (b), indicating the depth values and tag velocity, which were removed, since they do not add value to the image.

The annotation process was conducted using Label Studio². The annotations were exported with the YOLO annotation format given that YOLOv7 is the model chosen for this work. The YOLO annotation format consists of a .txt file associated with its corresponding image, where each line represents the annotation of an object in the image, meaning that an image may contain several annotated objects. The annotated object consists of 5 parameters: the object class, the coordinates (x,y) of the center of the bounding box, and the width and height values of the bounding box, all normalized according to the size of the image. Figure 3.3 shows two images with the bounding boxes around the animals identified.

The Jamanta dataset has 975 images with 634 annotated fish and 712 annotated jamanta. As a pre-processing step, the dataset was analyzed, which included separating the red, green and blue color channels for each image, and computing the mean of each channel for all images. Figure 3.4 shows how the mean values of each color channel is distributed throughout the dataset.

As expected for the Jamanta dataset, the blue channel is the most predominant one for all

¹<https://www.ffmpeg.org/>

²<https://labelstud.io/>



Figure 3.3: Example of annotated images

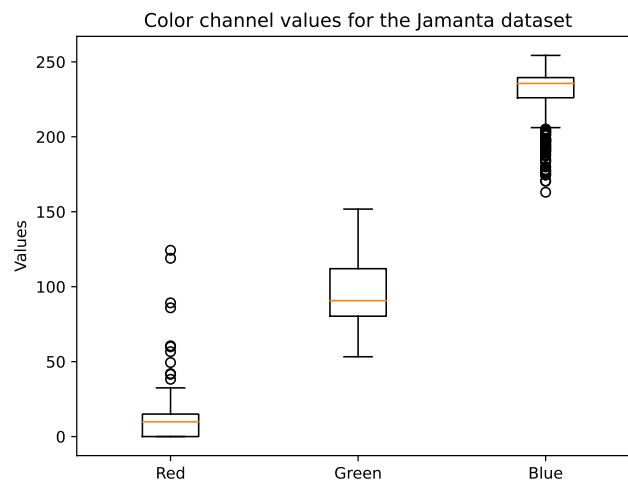


Figure 3.4: Boxplots with the intensities of each color channel (Red, Green, Blue) for the Jamanta dataset

the images of the dataset, and the red channel has very low values, even though some outliers are present. These outliers occur in images where the towed jamanta is near the surface, since light has not been significantly absorbed yet. The boxplots of the green channel show that the green color is always present in the images, especially when the animals are hardly identified. This prevalence of the green channel indicates that the jamantas and fish are mainly represented by the green channel, while the blue channel is associated with the background, and the red channel represents near-the-surface images.

The bounding box sizes were also considered, since the annotation indicates the position of the box on the image. Figure 3.5 shows boxplots of the bounding boxes width (a) and height (b) for the Fish and Jamanta classes.

Through this analysis, it was possible to assess the dimensions of each class, especially the fish since some instances were incorrectly annotated as jamantas and vice-versa.

Instead of selecting specific images for the validation dataset, a cross validation pipeline was

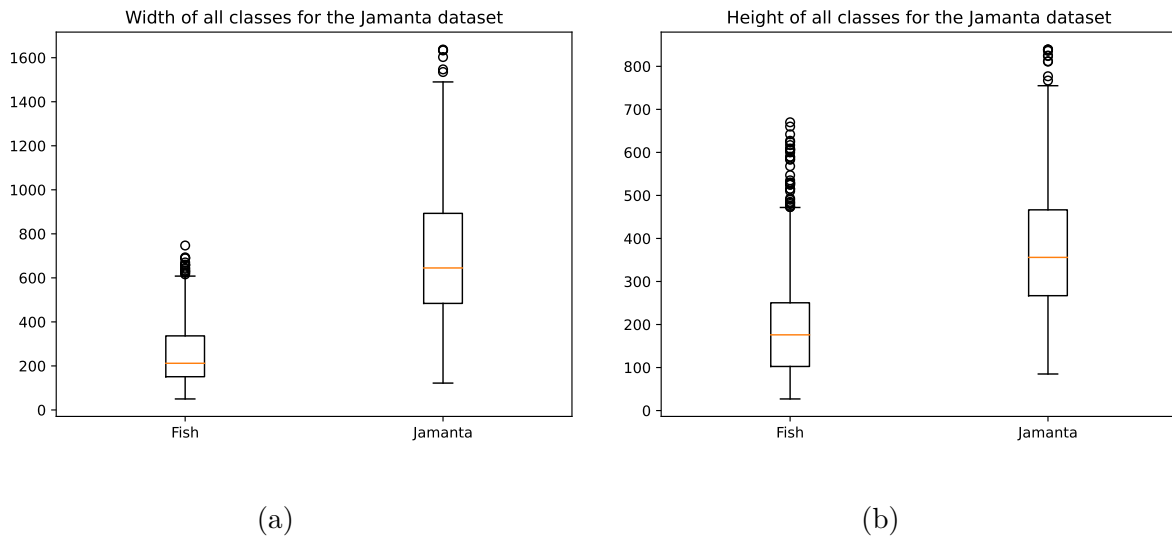


Figure 3.5: Boxplots with the width (a) and height (b) of the bounding boxes for the Fish and Jamanta class

developed to split the data into training and validation sets, where 85% of the total images were randomly assigned to the training set, and the remaining 15% were assigned to the validation set. In addition, a test set consisting of 84 images was created to provide additional insights and further evaluate the results.

3.2.2 Aquarium dataset

A dataset with 638 images was acquired from the Roboflow website³, with 4821 annotations across seven classes: Fish (0), Jellyfish (1), Penguin (2), Shark (3), Puffin (4), Stingray (5), Starfish (6). Figure 3.6 shows an example of an image from this dataset, where sharks and several fish are identified. The dataset splitting consisted of 448 images for the training set, 127 for the validation set and 63 for the test set.

The Aquarium dataset was considered mainly because of the need for high-quality and diverse underwater images. Since the images were taken from an aquarium, the recording environment is more predictable and controlled, and they are not as attenuated as the images taken from the ocean. Having said that, it has more diversity compared to the Jamanta dataset because of the total number of annotations, shown in Table 3.1. However, the Aquarium dataset shows serious class imbalance problems, since the Fish class is over-represented (having over 2600 annotations).

Through experimental training sessions with the Aquarium dataset, it was verified that the model behaved poorly, especially for the puffin class. For this reason, data augmentation techniques, such as horizontal flips and vertical flips, were applied only to this class. This way, the number of annotations of the Puffin class increased to 350 and the dataset's size increased to

³<https://public.roboflow.com/object-detection/aquarium>

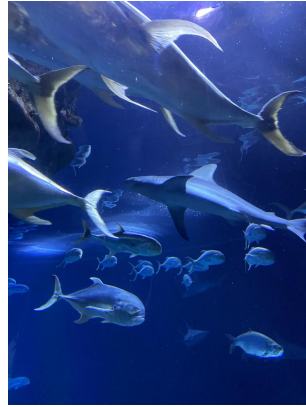


Figure 3.6: Example of images from the Aquarium dataset

Table 3.1: Distribution of the classes of the Aquarium dataset for the training, validation and test sets

Classes	Training	Validation	Test	Total
Fish	1961	459	249	2669
Stingray	136	33	15	184
Penguin	330	104	82	516
Shark	259	57	38	354
Puffin	175	74	35	284
Jellyfish	385	155	154	694
Starfish	78	27	11	116
Total	3324	909	584	-

683.

Furthermore, the Aquarium dataset was also considered since previous training experiments showed non-satisfactory results using other datasets, such as the msCOCO [44] dataset. The msCOCO was used during the experimental training sessions following the principles of transfer learning. This approach involves transferring the learnable parameters from previous training sessions to new domains, leading to faster and improved results [51]. However, the msCOCO does not include any class with marine animals, which will not generalize appropriately on an underwater setting. Therefore, the Aquarium dataset was considered, instead of the msCOCO, following the experiments with the msCOCO dataset. This approach guaranteed that the model learns the most suitable patterns for the application of this work.

Figure 3.7 shows the mean color channel values for the Aquarium dataset. Considering the aquariums that store the fish have different kinds of lighting, the color diversity of the images is expected. Additionally, the quality of the images is significantly better than the images from the Jamanta dataset, since the images from the Aquarium dataset are taken in a more controlled environment. Furthermore, Figures 3.8 shows the bounding box width and height values.

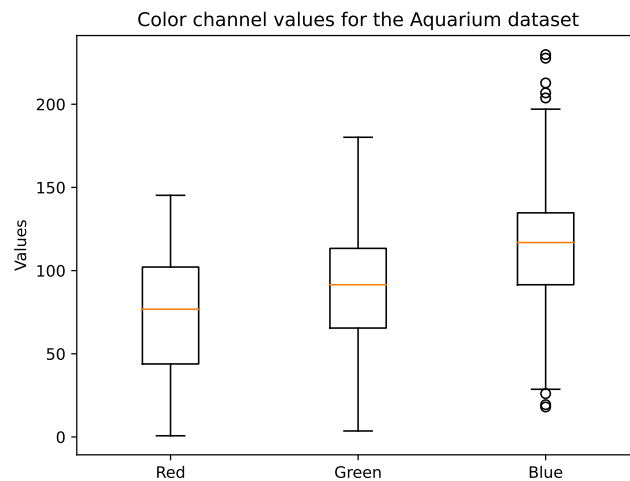


Figure 3.7: Boxplots with the intensities of each color channel (Red, Green, Blue) for the Aquarium dataset

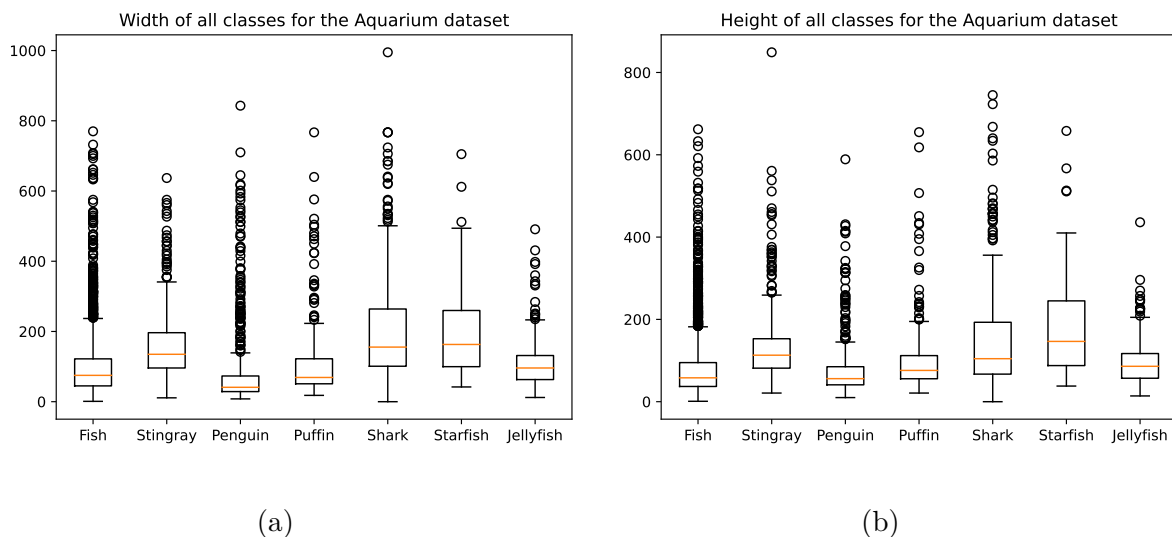


Figure 3.8: Boxplots with the width (a) and height (b) of the bounding boxes for all the classes of the Aquarium dataset

The boxplots from Figure 3.8 show that the sizes of the annotations of the Aquarium dataset are more unpredictable than the sizes of the annotations from the Jamanta. Considering the images were taken from an aquarium, the animals may be closer or farther from the camera, even if the animal itself is comparatively small. For example, sharks are usually larger than starfish but, in this case, their sizes are similar. This analysis confirms that the dataset is significantly diverse and a good starting point during the initial phases of model training.

3.3 Underwater Image Processing

As was discussed in Section 2.2.3, IP techniques may be used in underwater OD, to verify if enhancing the underwater images has a positive influence on the performance of an OD model. Even though previous studies [6, 19] noted that there is no strict correlation between enhancing the images and the performance of an OD model, it was decided that it should be verified for the development of this thesis. Therefore, two underwater IP methods were used to enhance the images from the Jamanta dataset. More specifically, the articles Xiang et al., 2018 [77] and Ghani & Isa, 2015 [78] were chosen to enhance the images from the Jamanta dataset.

Xiang et al., 2018 [77] proposed a method where the red channel is compensated using a linear combination of the green and blue channels, considering the attenuation coefficient of each color from the RGB color model. Then, guided filters were used to enhance the objects' contours, and a gamma correction model was proposed that stretched the histograms of the images to an appropriate configuration.

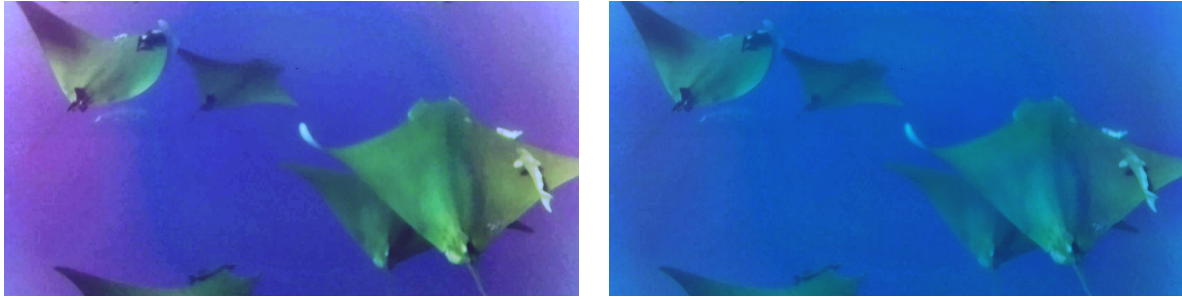
Ghani & Isa, 2015 [78] is an extension of two other image processing methods: ICM [79] and UCM [80], both based on histogram stretching of the RGB color model. UCM forces the distribution of the RGB histograms of the images to follow the Rayleigh distribution, which has been proven to be the most appropriate distribution for the histograms of underwater images [81] and is given by

$$F_{Rayleigh}(x) = \left(\frac{x}{\alpha^2}\right) e^{-\frac{x}{2\alpha^2}} \quad (3.1)$$

where $\alpha > 0$ is the parameter from the Rayleigh distribution and $x \geq 0$ is the intensity value.

However, Ghani & Isa, 2015 [78] extended the UCM method by converting the RGB image to the HSV color space, followed by stretching the histograms of the S and V components. Then the image is converted to the RGB color space to acquire the enhanced image. Both methods were applied to the Jamanta dataset since the attenuation issues were more evident than the issues from the Aquarium dataset. Figure 3.9 shows both methods applied to the original image, shown in Figure 3.2 (a).

Firstly, both methods reduce the dominance of the blue color. Additionally, the contrast is enhanced and the colors of the animals are restored. However, the Xiang et al. [77] method seems to add extra noise and accentuate colors that do not belong to the animals. Therefore, it appears that the processing conducted with the Ghani & Isa [78] method is more subtle, and more appropriate to the Jamanta dataset than processing with the Xiang et al. [77] method. To verify this, Figure 3.10 shows the boxplots of each color channel of the Jamanta dataset processed with the Xiang et al. [77] (a) and the Ghani & Isa [78] (b) methods, respectively. Figure 3.10 (a) shows that the Xiang et al. [77] method compensates the red channel for almost all the images. Comparing with Figure 3.4, the increase in the red color channel explains why some images processed with this method are over-enhanced. On the other hand, when comparing with Figure

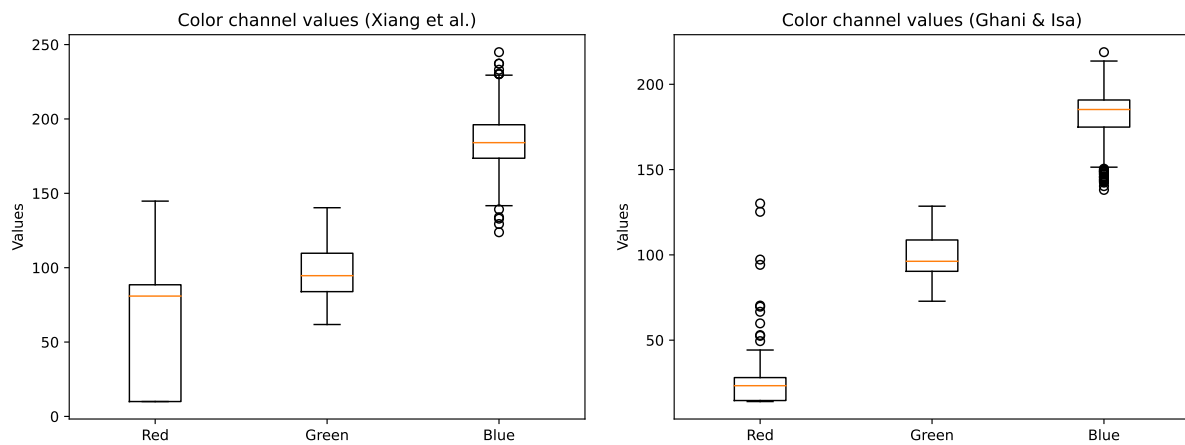


(a) Xiang et al. [77]

(b) Ghani & Isa [78]

Figure 3.9: Enhanced images applying the Xiang et al. [77] (a) and Ghani & Isa [78] (b) methods, applied to image from Figure 3.2 (a)

3.10 (b), the differences are minimal in the red color channel, while the values for the blue color channel are slightly reduced.



(a) Xiang et al. [77] method

(b) Ghani & Isa [78] method

Figure 3.10: Boxplots with the intensities of each color channel (Red, Green, Blue) for the Jamanta dataset processed with the Xiang et al. [77] method (a) and the Ghani & Isa [78] method (b)

To confirm that the Ghani & Isa's [78] enhancement is better than the Xiang et al. [77] method, Table 3.2 shows the results of applying the IQM presented in Table 3.2 to the reference and enhanced images. Since the results of the qualitative analysis vary from person to person, applying these metrics is crucial to decide which is the most appropriate enhancement, in an unbiased way. The results from Table 3.2 are obtained from computing the average of each metric for 13 images selected for the processing task.

Analyzing the entropy for each color channel, the red channel contains the least amount of information, especially for Reference, while the green channel contains almost all of the relevant information. By applying the Xiang et al. [77] method, the entropy increases significantly for

Table 3.2: Entropy of the (Red, Green, Blue) channels of the reference and enhanced images, with their respective MSE and PSNR values.

Methods	Entropy (R)	Entropy (G)	Entropy (B)	MSE	PSNR
Reference	1.987	5.744	4.916	-	-
Ghani & Isa [78]	2.535	5.618	5.699	977.87	18.32
Xiang et al. [77]	3.464	6.675	6.351	2776	14.15

each color channel, where the red channel increased the most. On the other hand, when applying the Ghani & Isa [78] method, the green channel's entropy decreases, while the entropy for the red and blue channels slightly increases. These results confirm that the Xiang et al. [77] method over-enhances the colors of the images. It also shows that processing with the Ghani & Isa [78] method is more subtle since it highlights the important information without compensating the image with non-existent colors.

The conclusions from analyzing the MSE and PSNR columns are also according to the conclusions from the entropy metric. The MSE values are considerably higher for the Xiang et al. [77] than the Ghani & Isa [78], while the PSNR is lower for the former. A decrease in PSNR is expected considering the relationship between PSNR and MSE.

Therefore, both quantitative and qualitative analyses confirm that the Ghani & Isa [78] method is the best IP method to use for the Jamanta dataset.

3.4 Technology used

MATLAB was used to process the images using the Image Processing Toolbox. Several Python libraries were used to obtain the results shown:

- Matplotlib: plot all the figures;
- Numpy: efficiently deal with arrays and matrices;
- Albumentations⁴: apply data augmentation techniques on image data
- YOLOv7 libraries: Tensorflow, Pytorch, OpenCV, Pillow, Scipy, pandas, scikit-learn

⁴<https://github.com/albumentations-team/albumentations>

3.4.1 Training the YOLOv7 model

To train the YOLOv7 model, the GitHub repository⁵ needs to be cloned to train the model on image data. This repository consists of several changeable properties, such as hyperparameters and architectures. The first training sessions and detections were done with the default settings, to understand how training is usually implemented. To guarantee the results are achieved in a timely manner, the GPU RTX A6000 was used.

During training, a set of arguments can be used with different values and combinations, such as hyperparameters, pre-trained weights, the number of training epochs, and many others. The number of epochs specifies the number of iterations that training lasts. In each epoch, the model verifies its results on the validation set, outputting the Precision and Recall, mAP@0.5, mAP@0.5:0.95. It also outputs each loss function value for the training dataset (Objectness, Classification, and Box), and the total loss value for the training dataset (Equation 2.9). The mAP@0.5 measures the mAP at an IoU threshold of 0.5. The mAP@0.5:0.95 measures the mean value of the mean Average Precision over the following IoU interval [0.5, 0.95] with a step of 0.05. Since mAP@0.5:0.95 uses several IoU thresholds, its value is expected to be smaller than mAP@0.5. Even so, mAP@0.5:0.95 is more robust than mAP@0.5 at assessing the object's localization accuracy.

Before training the model, pre-trained weights are an alternative, where the model has already learned features from previous training sessions, to improve the performance of the following training sessions. This approach follows the principles of transfer learning [51] and is systematically used in this work to guarantee the best training results. More specifically, YOLOv7 is first trained with the Aquarium dataset. Then, the weights (or learnable parameters) are used to train the Jamanta dataset. This way, the model learns the patterns with the Aquarium dataset, a high-quality underwater dataset, which generalizes appropriately to the Jamanta dataset.

To understand the best combination of hyperparameters, an alternative to the grid search methodology was carried out. Instead of training with all the configurations of the pre-specified grid, the best hyperparameters were chosen throughout training sessions, reducing the number of combinations needed to conduct this experiment. Therefore, BS, LR, WD, IR and Mom. were the chosen hyperparameters for the grid search experiment.

Finally, after a training session is complete, a folder is created with its outputs. More specifically, a .png file with the plots of the performance metrics. These plots contain the individual loss functions, i.e. Box, Classification, Objectness, for both training and validation datasets, the Precision and Recall metrics, the mAP@0.5 and the mAP@0.5:0.95. There is also a *weights* folder, containing a specific *best.pt* file, which is associated with the epoch where the mAP@0.5:0.95 achieves its maximum value. A confusion matrix is also one of the training outputs, which verifies the classification accuracy, by checking the TP, FN, FP and TN.

⁵<https://github.com/WongKinYiu/yolov7>

Chapter 4

Results and Discussion

In this chapter, the results of training YOLOv7 are presented, joined by a discussion of those results. The first section consists of selecting the best combination of the hyperparameters of YOLOv7. The purpose of this selection is to guarantee that the results are the most suitable for the data sets presented in Chapter 3. Then, the results of training YOLOv7 are shown.

4.1 Model Hyperparameter Selection

BS, LR, WD, Mom. and IR were the hyperparameters chosen to conduct this experiment, and their values are shown in Table 4.1.

Table 4.1: BS, LR, WD, Mom. and IR range values used for the subsequent analysis, where the bold values are the default values from YOLOv7.

Hyperparameters	Values
Batch Size (BS)	16,32,64
Learning Rate (LR)	10⁻², 10⁻³, 10⁻⁴, 10⁻⁵
Weight Decay (WD)	0, 10 ⁻³ , 5 × 10⁻⁴ , 10 ⁻⁴ , 10 ⁻⁵
Momentum (Mom.)	0.85, 0.9, 0.937 , 0.95, 0.99
Image Resolution (IR)	256, 512, 640 , 800

BS and LR are two of the most important hyperparameters of a neural network [82]. Previous experiments confirmed that varying these values significantly improves training performance.

Initially, $BS = \{2, 4, 8, 128, 256\}$ were chosen but training with $BS = \{2, 4\}$ showed unstable results, and training with $BS = \{128, 256\}$ was not possible due to a lack of computational memory to carry out these experiments. $BS = 8$ showed similar results to those values from Table 4.1, differing mainly on training time, which is why it was excluded from this table. This

way, the number of training sessions slightly reduced.

To investigate the impact of LR variations on training, the LR values considered in Table 4.1 were derived from the default LR of 0.01 used in YOLOv7 for custom data sets. The LR values were multiplied by a factor of 0.1 at the start of each training session. Previous experiments with $LR = 0.1$ showed unstable results, while $LR = 10^{-6}$ required a longer training time to converge to an acceptable result, without a guarantee that it would happen. Therefore, $LR = \{10^{-1}, 10^{-6}\}$ were not used.

Furthermore, the WD values were considered by following the instructions in [82], which stated that smaller data sets require large WD values. Mom. values were also considered from the same paper, since it stated that it is valuable to set Mom. as large as possible. Finally, the effects of varying IR on model performance are also studied, since previous experiments showed significant changes on model performance.

Table 4.2 shows the mAP@0.5:0.95 values when training YOLOv7 with the Aquarium and Jamanta data sets with varying BS and IR and fixing LR, Mom. and WD with the default values from Table 4.1. The highest values are shown in bold, and the **E** column represents the epoch at which the mAP value was obtained.

Table 4.2: Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations

		Aquarium data set		Jamanta data set	
BS	IR	mAP@0.5:0.95 (%)	E	mAP@0.5:0.95 (%)	E
16	256	45.46 ± 0.09	244	72.8 ± 0.09	119
	512	50.12 ± 0.09	150	72.4 ± 0.1	199
	640	49.86 ± 0.09	100	71.79 ± 0.1	194
	800	49.37 ± 0.08	100	65.88 ± 0.2	278
32	256	45.38 ± 0.1	162	72.62 ± 0.1	129
	512	49.25 ± 0.09	222	73 ± 0.1	163
	640	49.42 ± 0.09	111	70.1 ± 0.2	235
	800	48.63 ± 0.09	100	67.53 ± 0.2	260
64	256	44.14 ± 0.1	232	72.4 ± 0.1	139
	512	49.48 ± 0.1	186	71.53 ± 0.1	113
	640	50.61 ± 0.1	230	72.75 ± 0.2	155

Changing the IR with the BS does not alter significantly the mAP values, except for the Aquarium data set which is slightly lower when $IR = 256$. Because of this, a fixed IR of 512 was considered for the following training sessions, reducing the amount of combinations needed to obtain the best combination of hyperparameters.

Even though the mAP values are similar, the main difference from each training session is the

time and computational memory it takes to train the model. Higher IR values (greater than or equal to 640×640) lasts almost one hour longer than smaller IR values (lower than 640×640) and the memory also drastically increases. Given that large BS values decrease the computation time, an appropriate experiment would be to increase BS as IR increases. However, as was the case for $BS = 64$ and $IR = 800$, training with higher values of BS and IR is not possible, due to lack of computational memory.

For the Aquarium data set, Table 4.2 shows that the $mAP@0.5:0.95$ is lower with lower IR values and higher for higher IR values. On the other hand, an opposite behavior is shown once the model is trained with the Jamanta data set. These differences may be explained because of the differences between the sizes of the annotations for each data set. More specifically, Figure 3.8 shows the boxplots of the bounding boxes' width and height values for the classes of the Aquarium data set. It is observed that the median width and height values are generally lower than 200 pixels. Once the images are compressed to a lower spatial resolution at the beginning of the training sessions, the detection task becomes harder for the model.

When analyzing the boxplots from Figure 3.5 for the Jamanta data set, it is observed that the median width and height of the bounding boxes for the Jamanta class are noticeably distinct from the bounding box sizes of the Aquarium data set. This indicates that, even after compression, the sizes of the jamantas remain sufficiently large for the model to detect effectively. Additionally, the Fish class within the Jamanta data set exhibits a similar bounding box size to the Aquarium classes (Figure 3.8). As a result, when compression occurs, the Fish class is smaller and the model may encounter difficulties in accurately locating this class. Nonetheless, when training the model with the Jamanta data set and smaller IR values, the results obtained become more acceptable. However, these results may be influenced by the positive detections involving the Jamanta class, introducing a bias in the evaluation process.

Therefore, varying IR at the beginning of the training sessions influences the performance of the model, especially if the classes of the data sets have small width and height values. Because of this, from this point on, the most appropriate IR is 512×512 .

Table 4.3 shows the $mAP@0.5:0.95$ values when training YOLOv7 with the Aquarium and Jamanta data sets with varying BS and LR, fixing Mom. and WD with the default values from Table 4.1 and $IR = 512 \times 512$. The highest values are shown in bold, and the **E** column represents the epoch at which the mAP value was obtained.

Across all BS and LR values, the mAP values did not alter significantly, except for $LR = 10^{-5}$. This was expected since low LR values mean that the network slowly updates its weights while the model is training. As shown by column **E**, the maximum mAP value is achieved at a higher epoch as the LR gradually decreases. Therefore, more training epochs are necessary to achieve higher mAP.

However, Figure 4.1 shows how the mAP values (a) and classification loss for the validation set (b) varies throughout all training sessions for $BS = 16$. By decreasing the LR, the classification loss gradually increases. For $LR = 10^{-5}$ (red), the loss stabilizes below 0.030 with no signs of

Table 4.3: Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations

		Aquarium data set		Jamanta data set	
BS	LR	mAP@0.5:0.95 (%)	E	mAP@0.5:0.95 (%)	E
16	10^{-2}	49.96 ± 0.07	127	73.6 ± 0.1	122
	10^{-3}	49.46 ± 0.1	156	72.93 ± 0.1	75
	10^{-4}	47.93 ± 0.1	376	72.62 ± 0.2	398
	10^{-5}	17.72 ± 0.07	388	30.2 ± 0.09	397
32	10^{-2}	50.02 ± 0.09	152	71.66 ± 0.1	114
	10^{-3}	49.98 ± 0.1	150	73.89 ± 0.1	134
	10^{-4}	47.63 ± 0.2	394	72.16 ± 0.2	313
	10^{-5}	17.36 ± 0.07	395	35.61 ± 0.1	399
64	10^{-2}	49.1 ± 0.1	327	69.06 ± 0.1	50
	10^{-3}	49.64 ± 0.1	236	73.45 ± 0.2	122
	10^{-4}	46.48 ± 0.2	397	75.9 ± 0.2	287
	10^{-5}	16.52 ± 0.06	396	56.18 ± 0.2	397

decreasing. This is also shown in Figure 4.1 (a) where the red curve is slowly converging to a value near 0.2, with no signs of stabilizing in a more satisfactory value. Therefore, $LR = 10^{-5}$ should not be considered for the Aquarium data set.

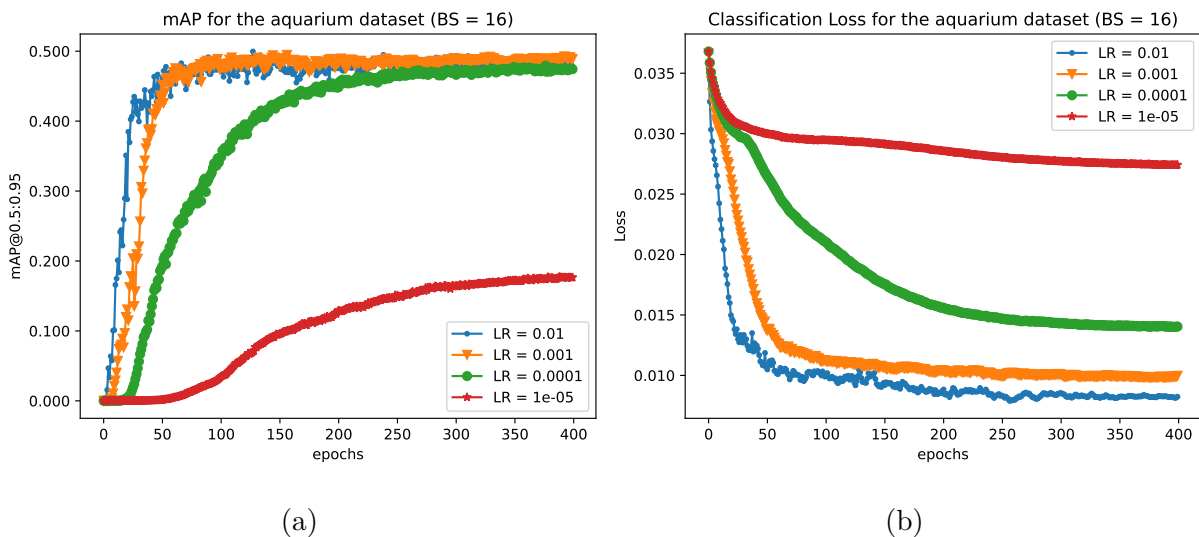


Figure 4.1: Plots of mAP@0.5:0.95 values (a) and classification loss for the Aquarium’s validation set (b) for BS = 16 and proposed LR values.

Once the LR is sufficiently high, training becomes unstable and may never converge to an

optimum result, which is why $LR = 10^{-1}$ was not considered for these experiments. However, the validation loss functions for $LR = \{10^{-2}, 10^{-3}\}$ for the Aquarium data set, start to increase at one point during training, indicating that the model may be overfitting the data, as shown in Figure 4.2. Once $LR = 10^{-4}$, the loss function continually decreases, suggesting that it is the most appropriate LR value for the Aquarium data set.

Figure 4.2 only shows the results for $BS = 64$ and $LR = \{10^{-2}, 10^{-3}, 10^{-4}\}$. When $LR = 10^{-5}$, the model behaves poorly, which is why it was excluded from this plot. When $BS = 32$, the plot of the objectness loss for the Aquarium data set behaves in a similar way, meaning that $BS = \{32, 64\}$ may be used across subsequent training sessions. Since increasing BS also increases the computational memory necessary to process the data, the choice between both BS values depends on the amount of training data. Nevertheless, $BS = 64$ was chosen for the following training sessions.

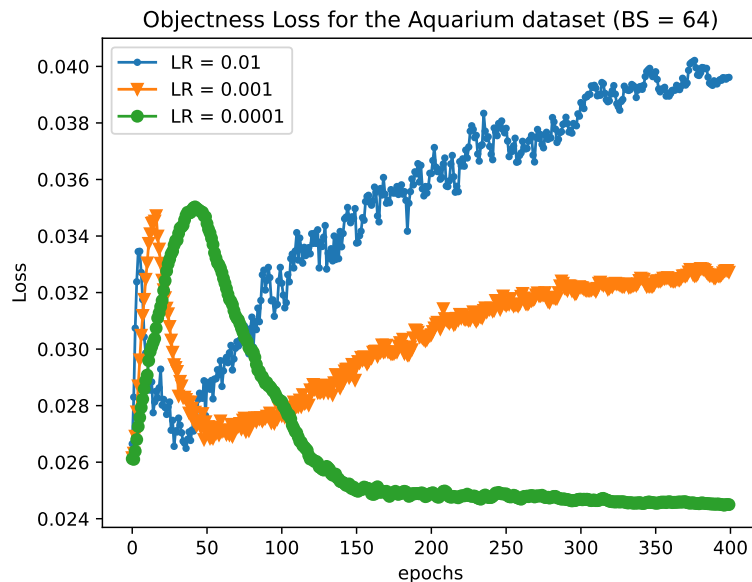


Figure 4.2: Plot of the objectness loss for the Aquarium’s validation set with $LR = \{10^{-2}, 10^{-3}, 10^{-4}\}$ and $BS = 64$.

The model was also trained with the Jamanta data set and the results were significantly better, reaching almost always mAP values larger than 70%. This is a satisfying result having in mind that this set is small and the quality of the images suffer from serious attenuation problems. Furthermore, the model’s performance with the Jamanta data set is similar to the performance with the Aquarium data set for all BS values, except for $BS = 64$. Fixing $LR = 10^{-5}$ and $BS = \{16, 32\}$, the mAP values for the Jamanta data set are below 40%. When $BS = 64$ for the same LR, the mAP value increases significantly. This behavior is better perceived from Figure 4.3 (a), where the mAP is still increasing during the 400th epoch, while other LR values slowly decay or maintain after reaching their maximum value in earlier epochs. However, Figure 4.3 (b) shows how the classification loss for the Jamanta’s validation data set varies for $BS = 64$, across all LR values. When $LR = 10^{-5}$, the classification loss stabilizes near 0.01, while the other LR

values converge between 0.002 and 0.004.

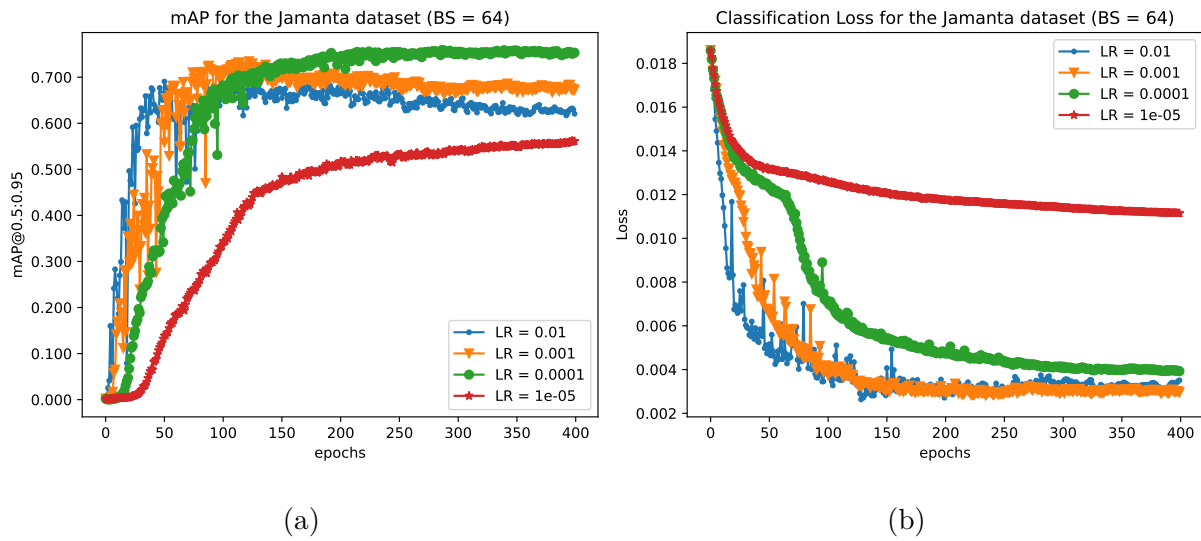


Figure 4.3: Plots of mAP@0.5:0.95 values (a) and classification loss (b) for the Jamanta's validation set for BS = 64 and LR = $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$

Additionally, the objectness loss for the Jamanta's validation data set behaves similarly to the objectness loss for the Aquarium's validation data set. This can be seen in Figure 4.4, where both LR = $\{10^{-2}, 10^{-3}\}$ increase at one point, while for LR = 10^{-4} , the loss decreases, and stabilizes near a loss value of 0.012.

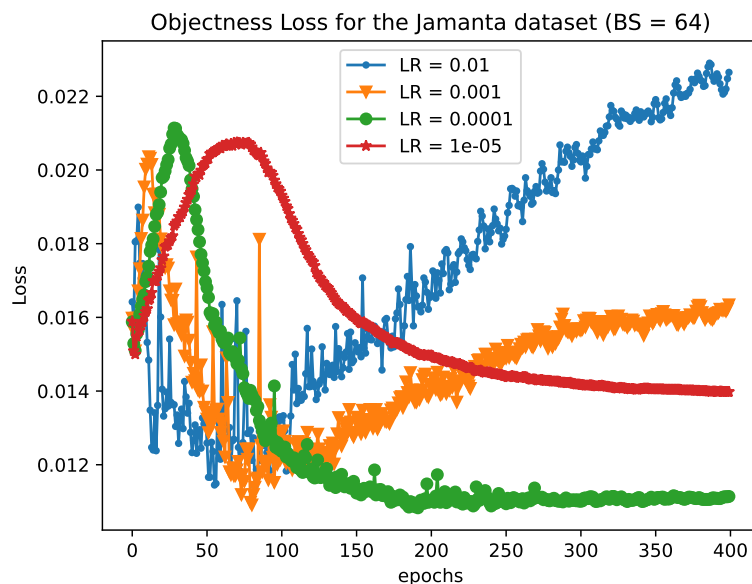


Figure 4.4: Plot of the objectness loss for the Jamanta's validation set with LR = $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ and BS = 64.

Since the results were similar for both Aquarium and Jamanta data sets, $LR = \{10^{-2}, 10^{-3}\}$ should not be considered because of an increase in the objectness loss functions. On the other hand, $LR = 10^{-5}$ achieves high loss values with no guarantees of converging to a more optimal value if more epochs are considered. When $LR = 10^{-4}$, the loss functions behave appropriately, meaning that the best results for the Jamanta data set occurs when $LR = 10^{-4}$ and $BS = 64$.

Therefore, for the following training sessions, $LR = 10^{-4}$ and $BS = 64$ are fixed. This way, there is a significant reduction in the amount of combinations necessary to understand how LR and BS behave with varying WD and Mom..

Table 4.4: Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations

WD	Mom.	Aquarium data set	E	Jamanta data set	E
		mAP@0.5:0.95 (%)		mAP@0.5:0.95 (%)	
0	0.85	42.43 ± 0.143	494	72.95 ± 0.167	449
	0.90	45.91 ± 0.157	471	74.38 ± 0.169	489
	0.937	48.64 ± 0.167	489	75.12 ± 0.170	392
	0.95	48.98 ± 0.169	481	74.68 ± 0.169	365
	0.99	50.36 ± 0.175	238	74.75 ± 0.157	132
10^{-3}	0.85	42.27 ± 0.143	477	72.89 ± 0.170	464
	0.90	45.94 ± 0.157	487	73.56 ± 0.169	391
	0.937	48.4 ± 0.167	493	73.71 ± 0.167	320
	0.95	48.88 ± 0.169	373	73.17 ± 0.164	201
	0.99	50.74 ± 0.176	323	73.91 ± 0.156	127
10^{-4}	0.85	42.4 ± 0.142	477	72.03 ± 0.168	453
	0.90	45.63 ± 0.157	487	72.67 ± 0.169	366
	0.937	48.45 ± 0.167	493	73.21 ± 0.168	230
	0.95	48.65 ± 0.169	373	72.92 ± 0.168	280
	0.99	50.66 ± 0.176	323	73.45 ± 0.158	128
5×10^{-4}	0.85	42.35 ± 0.140	494	72.65 ± 0.169	401
	0.90	45.71 ± 0.156	490	73.34 ± 0.169	334
	0.937	48.17 ± 0.166	490	73.15 ± 0.167	220
	0.95	48.83 ± 0.169	471	73.48 ± 0.166	234
	0.99	50.44 ± 0.174	312	73.9 ± 0.157	183
10^{-5}	0.85	42.46 ± 0.143	476	72.55 ± 0.169	487
	0.90	45.77 ± 0.156	490	73.35 ± 0.168	363
	0.937	48.63 ± 0.167	487	73.67 ± 0.168	262
	0.95	48.71 ± 0.169	489	73.52 ± 0.166	247
	0.99	50.25 ± 0.173	329	74.45 ± 0.156	127

Analyzing Table 4.4, showing the mAP@0.5:0.95 values when training YOLOv7 with the

Aquarium and Jamanta data sets with varying WD and Mom. and fixing $LR = 10^{-4}$, $BS = 64$ and $IR = 512 \times 512$, the Mom. hyperparameter has a major influence on the performance of the model. When training the model with the Aquarium data set, the mAP increases as Mom. increases. Training time is also reduced with an increase in Mom., confirming that this hyperparameter increases convergence speed and accuracy. However, the model behaves differently for the Jamanta data set, since higher Mom. does not guarantee higher mAP. Additionally, the Mom. hyperparameter appears to have more influence when training with the Aquarium data set, than training with the Jamanta data set.

By analyzing Table 4.4, it seems that choosing Mom. = 0.99 is the most appropriate choice. However, Figure 4.5 shows how the objectness loss for the Aquarium (a) and Jamanta (b) validation sets changes with Mom. = {0.85, 0.90, 0.937, 0.95, 0.99} and WD = 0. When Mom. = 0.99, the objectness loss increases after the 100th epoch for both data sets. This shows that the model may be overfitting the data for this Mom. value. For this reason, the results when Mom. = 0.99 should not be taken into consideration for the rest of the analysis. Considering the results were especially poor for the Aquarium data set when Mom. = {0.85, 0.90}, these values are also dismissed from this analysis. Therefore, both Mom. = 0.937 and Mom. = 0.95 are the most appropriate Mom. values to train the YOLOv7 model.

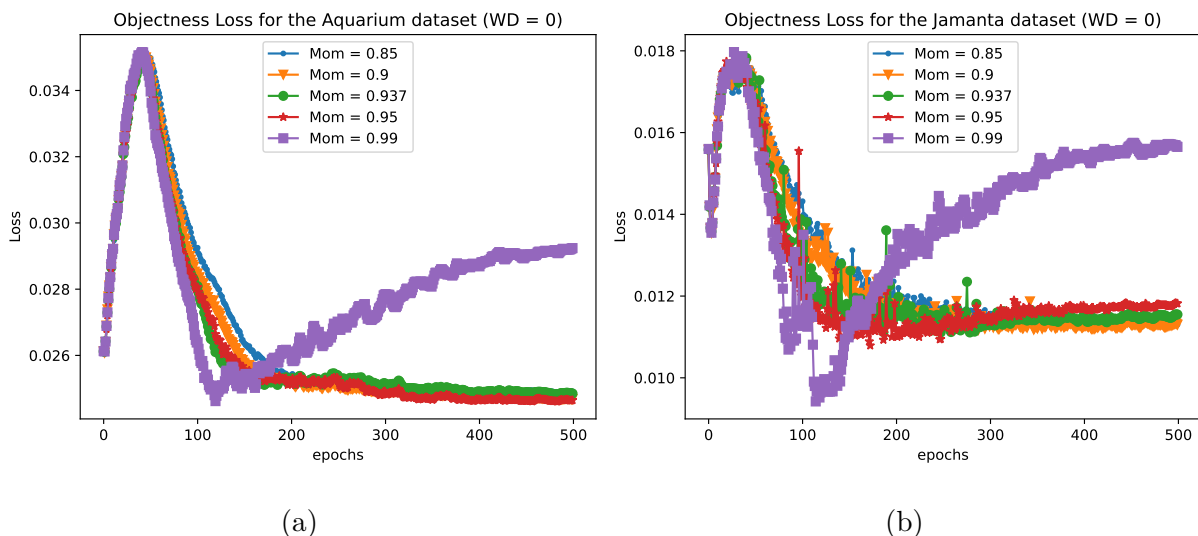


Figure 4.5: Plots of the objectness loss for the Aquarium’s (a) and the Jamanta’s (b) validation sets for WD = 0 and Mom. = {0.85, 0.90, 0.937, 0.95, 0.99}

Regarding the WD hyperparameter, Table 4.4 does not show significant differences when WD decreases, except for WD = 0 and Mom. = 0.937. This is a particular case since WD = 0 means there is no regularization technique applied to the loss function. Because of this, the model is more susceptible to overfitting problems. However, these issues did not occur, and the results without WD slightly improved the performance of the model. OD models include other regularization techniques, such as data augmentation techniques, batch normalization and dropout [67, 83]. These techniques are also applied to avoid overfitting and underfitting issues.

The fact that $WD = 0$ does not offer significant changes to the performance of the model shows that the other regularization techniques are effective enough to avoid those issues. Additionally, when analyzing the loss and mAP plots for the other WD values, this fact also verifies, since the model behaves similarly throughout all WD values. Therefore, $WD = 5 \times 10^{-4}$ is chosen as the best WD value. This way, it is guaranteed that the model does not suffer from overfitting problems.

For these reasons, Table 4.5 shows the best combination of hyperparameters for the Aquarium and Jamanta data sets.

Table 4.5: Values for BS, LR, WD, Mom. and IR selected

Hyperparameters	Values
BS	64
LR	10^{-4}
WD	5×10^{-4}
Mom.	0.937
IR	512×512

4.2 Training Results and Discussion

After choosing the best set of hyperparameters, it is possible to interpret the results from training YOLOv7. The system outputs several plots, to understand how the model behaved. The results are shown in Appendix A. This section explains and discusses the results of training YOLOv7 with the Aquarium and Jamanta validation and test sets. Both validation and test sets are evaluated since the validation set is used during training to understand how the model behaves and to tune the model’s parameters. With the test set, it is possible to verify how the trained and validated model would behave in a situation close to a real-world application. Therefore, the results from the test set give an important insight into the applicability of the model.

Starting with the results for the Aquarium’s validation data set, Figure 4.6 shows the confusion matrix for all the classes of the Aquarium data set. The model appears to be accurate in classifying all the classes, since it detects the correct class over 80% of the time, except for the Puffin class. The amount of FN for the Puffin class shows that the model does not detect 30% of the time. Additionally, even though YOLOv7 correctly identifies most of the objects as fish, the amount of FP for this class is still undesirable. The fact that the FP for the Fish class is almost 50% means that the model is constantly detecting this class. Therefore, either the model is correctly detecting fish that were not previously annotated, or it is labeling objects as fish that are background objects. Either way, constantly looking for a class is not ideal since most of the detections can be inaccurate, i.e. the amount of FP is high.

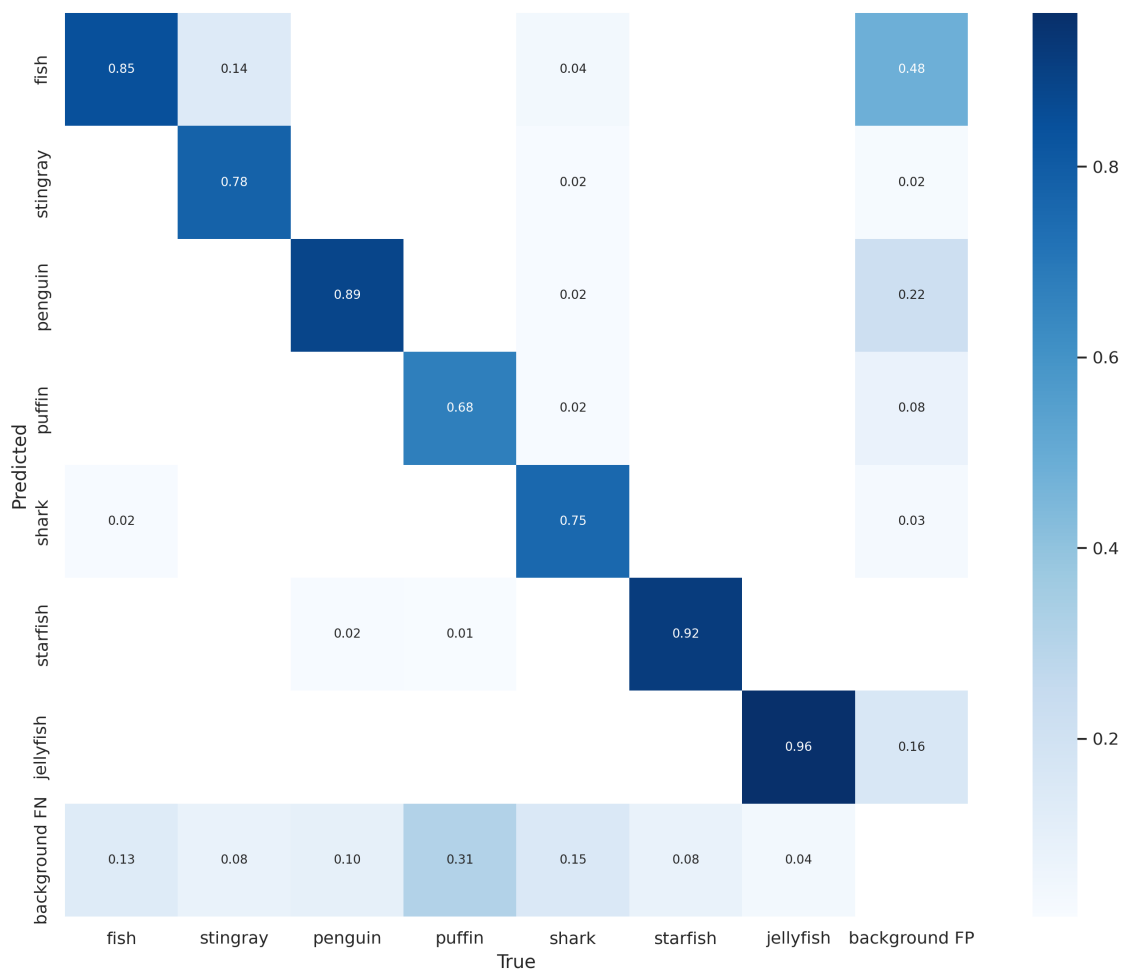


Figure 4.6: Confusion matrix for all the classes of the Aquarium's validation data set

The precision and recall (Equations (2.5) and (2.6), respectively) can be calculated from the confusion matrix, and the amount of FP and FN influences these metrics. Figure 4.7 and Figure 4.8 shows the precision and recall plots for the classes of the Aquarium data set over increasing confidence thresholds, respectively. These thresholds are related to the confidence of the bounding boxes generated by YOLOv7 [53].

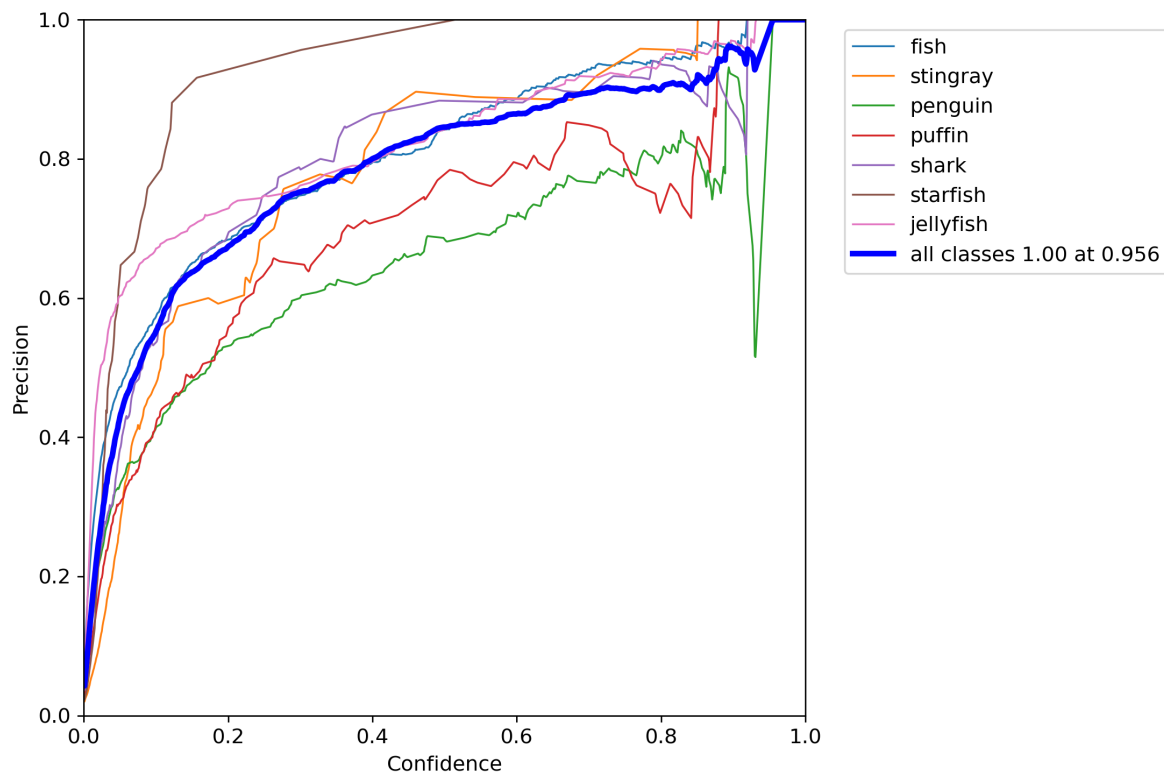


Figure 4.7: Plot for the Precision curve measured over increasing confidence thresholds for the Aquarium’s validation data set

The precision plot (Figure 4.7) shows that as confidence increases, the precision also increases for all the classes of the Aquarium data set. An increase in precision means that the number of FP decreases and/or the number of TP increases. Either way, the model has more confidence in accurately locating the objects, given that the amount of positive predictions increases. Furthermore, all classes of the Aquarium data set reach their maximum precision at confidence thresholds larger than 0.8, except for the Starfish class. This shows that YOLOv7 most likely detects a starfish without being certain that it is a starfish. A solution for this issue could be to acquire more data labeled as starfish and verify how this changes the model’s performance.

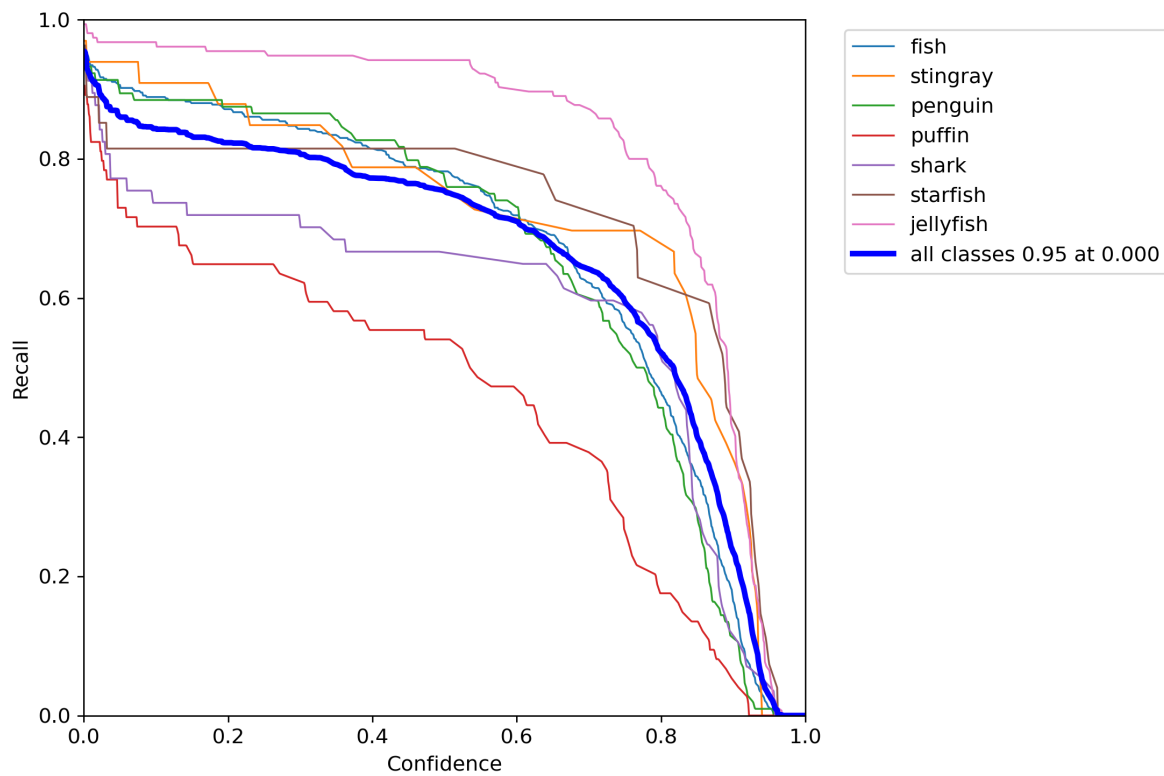


Figure 4.8: Plot for the Recall curve measured over increasing confidence thresholds for the Aquarium's validation data set

The recall plot (Figure 4.8) shows that as confidence increases the recall decreases for all the classes of the Aquarium data set. An increase in recall means that the number of FN decreases and/or the number of TP increases. Either way, the model appears to have more confidence over its predictions, even though most may be incorrect. Therefore, the model's confidence decreases since more objects are classified, with no guarantees that the prediction is correct. Analyzing Figure 4.8, only the Puffin class behaves differently from the other classes of the Aquarium data set, since it is the class with the lowest recall, at all confidence thresholds. This difference is explained by the amount of FN that the Puffin class has. Since FN is high, the recall for this class decreases.

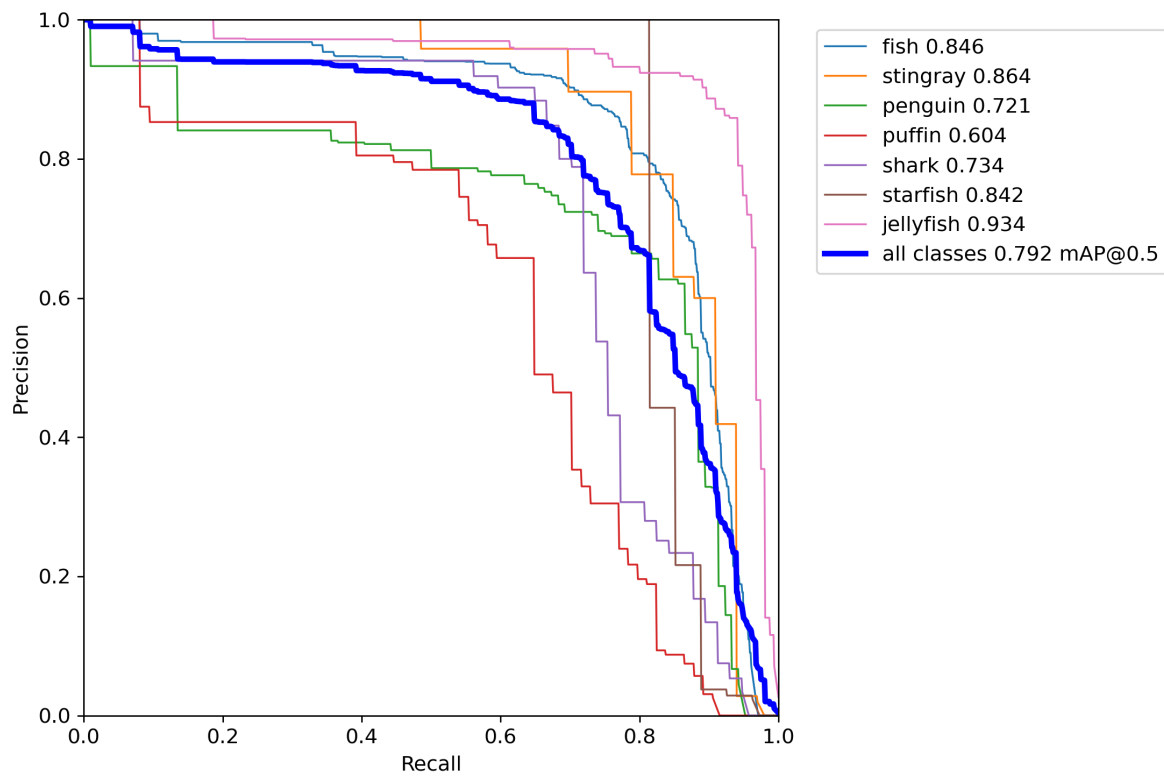


Figure 4.9: Plot for the PRC curve measured over increasing confidence thresholds for the Aquarium’s validation data set

Additionally, Figure 4.9 shows the PRC. An OD model is ideal if the blue line intersects the upper right corner. However, this is impossible since, generally, an increase in precision implies a decrease in recall and vice-versa. Nevertheless, a good OD model is obtained once the precision stays high as its recall increases [53]. Figure 4.9 shows the relationship between the precision and recall metrics, showing exactly that as one increases, the other decreases. Calculating the AUC of the PRC computes the mAP@0.5 for all the classes of the data set, which is an important indicator over the model’s performance. The results for each class were acceptable, despite the 60% mAP for the Puffin class. Both the precision (Figure 4.7) and the recall (Figure 4.8) already show this for the Puffin class, where the model is usually precise when identifying this class, having low FP values (i.e. high precision). However, due to high FN values, the recall is smaller, meaning that the model does not locate this class most of the time. To fix this issue, data augmentation techniques are an alternative.

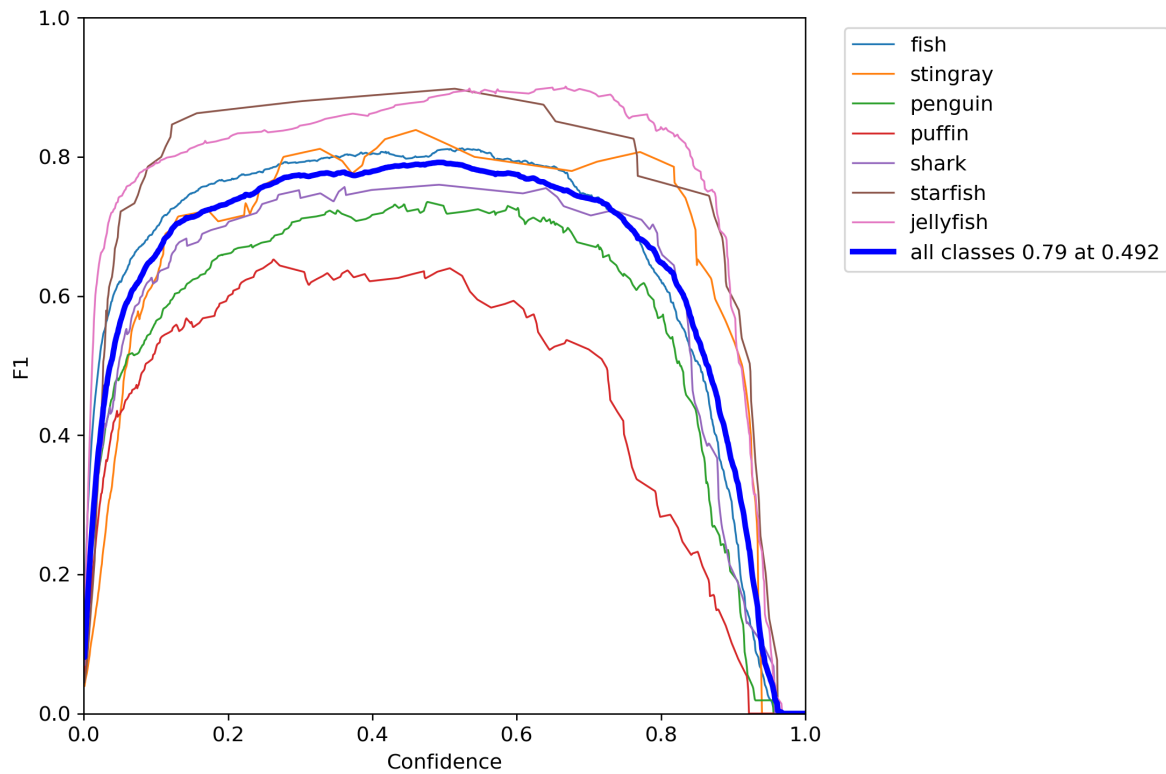


Figure 4.10: F1-curve measured over increasing confidence thresholds for the Aquarium’s validation data set

The mAP values are high enough to conclude that the model performed well with the Aquarium data set. However, the F1 score is another important metric that is especially used to evaluate the performance of an OD model on imbalanced data sets. Therefore, Figure 4.10 shows the F1-score curve for the Aquarium’s validation data set over increasing confidence thresholds. Since the precision and recall are both functions of the confidence thresholds, the F1-score also varies accordingly (Equation (2.2.1)). The F1-score of all the classes, represented by the blue line in Figure 4.10, shows acceptable results. Only the Puffin and the Starfish classes behave differently. The Puffin class has lower F1-score values than the other classes. Considering both precision and recall are lower for the Puffin class, it is clear that the F1-score is also low. On the other hand, the Starfish class shows exceptionally good results. The confusion matrix for the Aquarium’s validation set (Figure 4.6) shows that 8% of the predictions are FN, 0% are FP, and 92% are TP for the Starfish class, explaining why the F1-score is as high for this class. Nevertheless, the model behaving well for the Starfish class is not expected, considering the number of starfish annotations (116 in total).

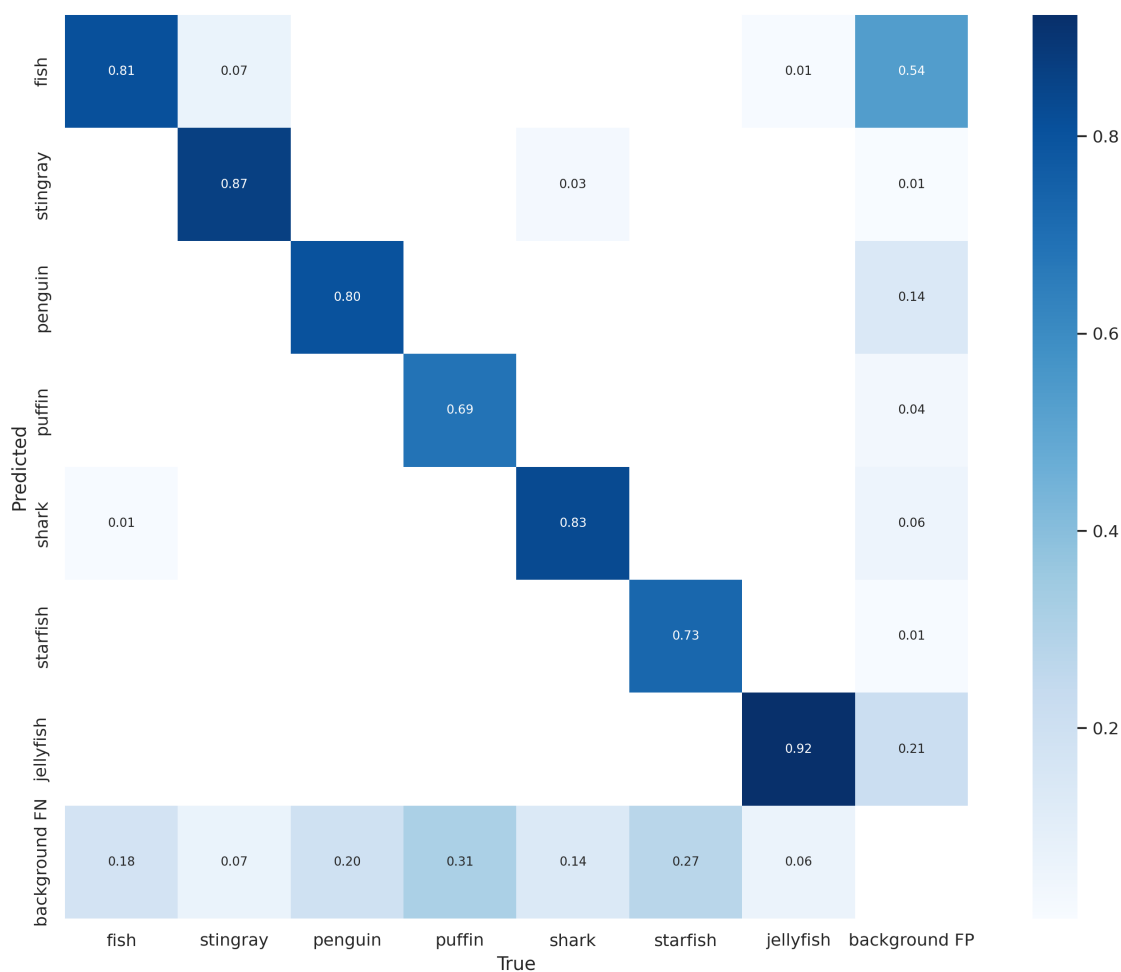


Figure 4.11: Confusion matrix for all the classes of the Aquarium’s test data set

Furthermore, the test set is used to evaluate the results from training YOLOv7 with the Aquarium’s validation data set. Figure 4.11 shows the confusion matrix for the Aquarium’s test set. Because of the test set, it is possible to verify if the results for all the classes are accurate, or if YOLOv7 overestimated them during training.

Generally, the results from training with the Aquarium’s validation set are better than training with the test set, especially for the Stingray class. However, the Starfish class shows a significant decline. More specifically, the model does not detect almost 30% of the time when a starfish is present, leading to a decrease in the recall (Figure A.2). For the Stingray class, there was an increase in the number of TP with the test set, since this class is occasionally detected as fish for the validation data set. Since both precision and recall increase, the F1-score also increases for the Stingray class of the Aquarium’s test set. The fact that the results for the validation set are better than the results for the test set indicates that the model may be overfitting the validation set. Therefore, the model’s generalization power may not be enough to apply on unknown videos. To deal with these issues, more data is required, which can be obtained from new images or by applying data augmentation techniques.

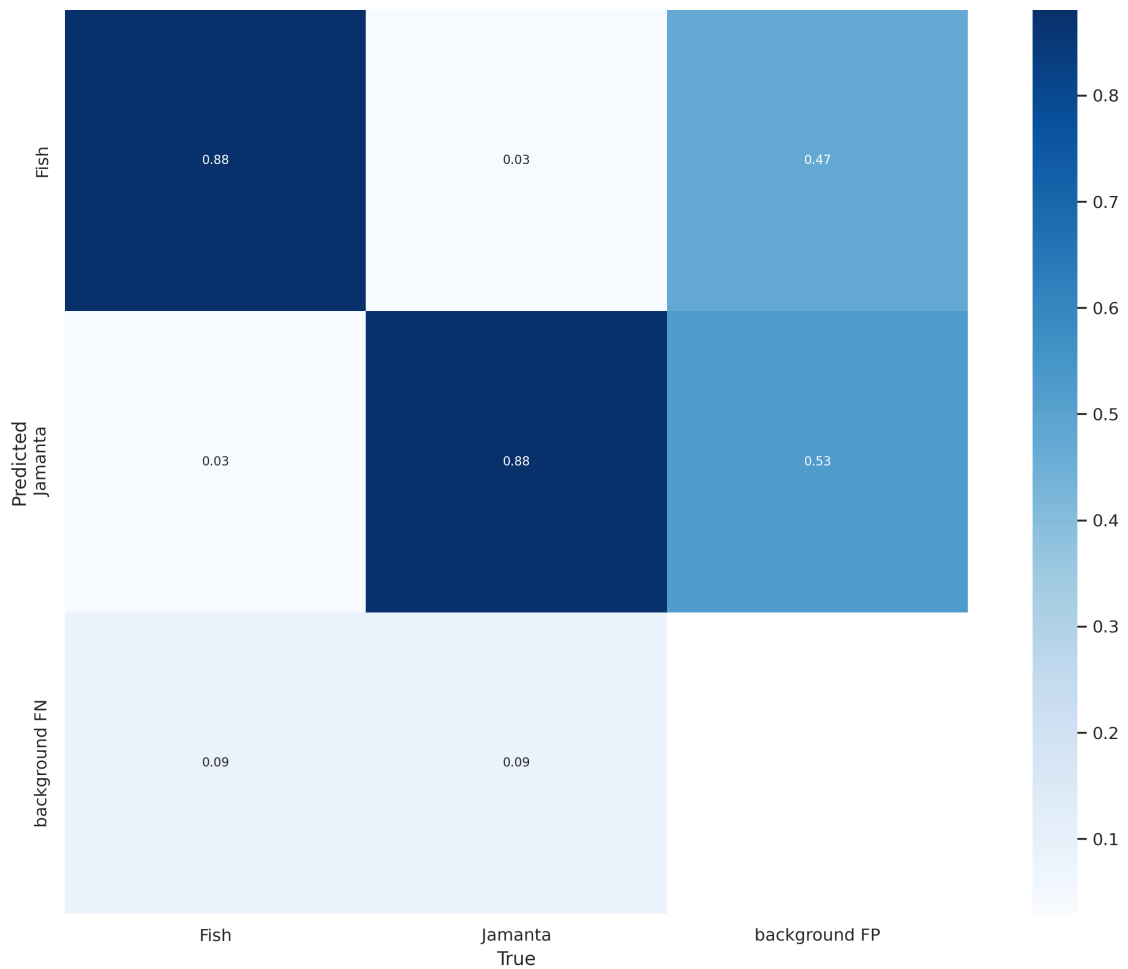
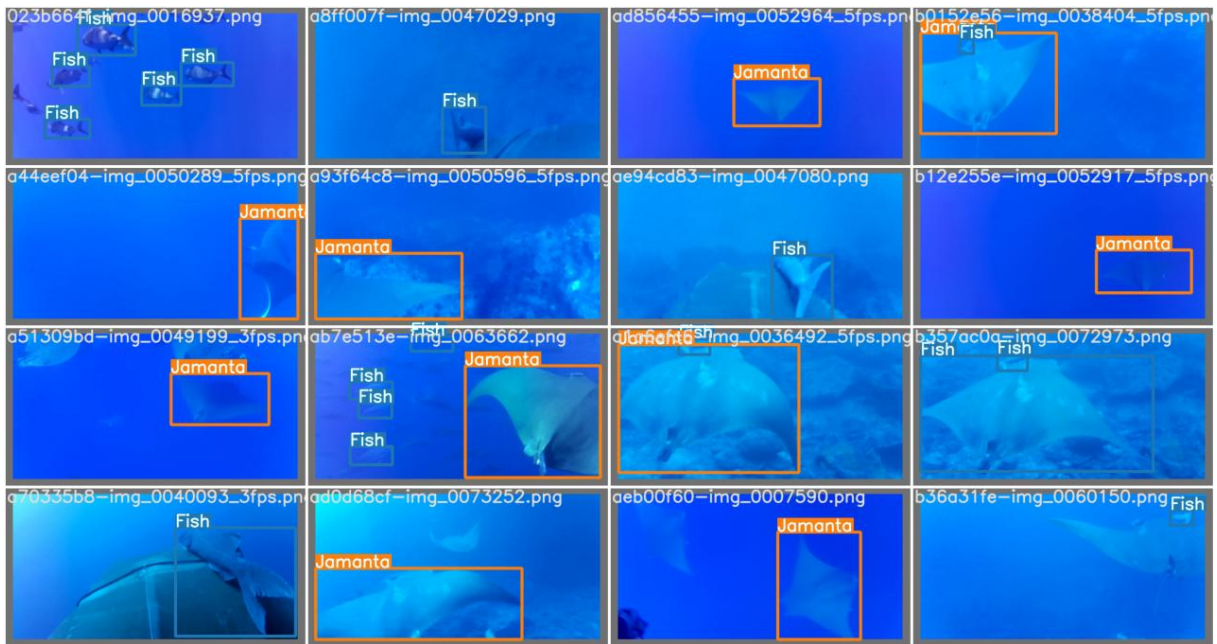
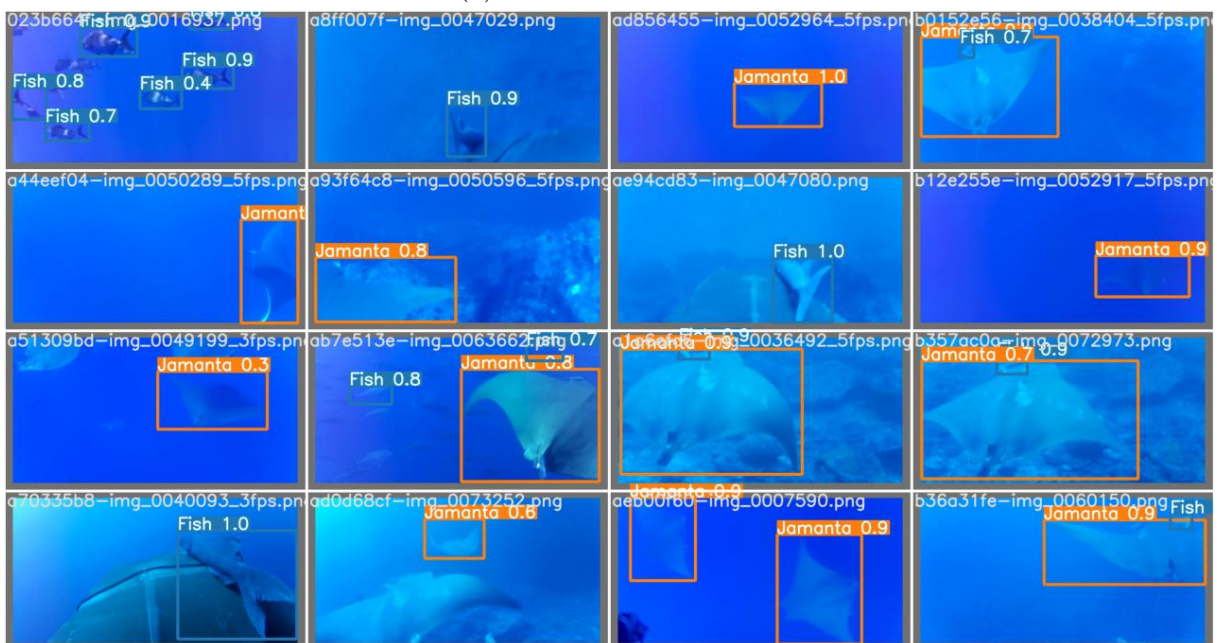


Figure 4.12: Confusion matrix for all the classes of the Jamanta’s validation data set

Regarding Jamanta’s validation data set, Figure 4.12 shows the confusion matrix when YOLOv7 was trained with Jamanta’s validation data set. As was the case with the Fish class for the Aquarium sets, the Fish and Jamanta classes deal with the problem of having FP values close to 50%. As a consequence, the model is either correctly classifying the animals or it is identifying objects that are background objects. To confirm this, YOLOv7 also outputs a mosaic of images where the ground truth and the predictions are compared. These images are presented in Figure 4.13, showing that the model detects several instances where a jamanta is present, while not being annotated in the validation data set. This means that the detections performed by the model are true detections for both Fish and Jamanta classes, explaining why the FP is so high. Having said that, the annotation process of the Jamanta data set should be revised, to guarantee that most objects are identified. This way, it is possible to check how the FP varies.



(a) Ground-truth labels



(b) Model's predictions

Figure 4.13: Ground-truth labels (a) and model's predictions for the Jamanta's validation data set

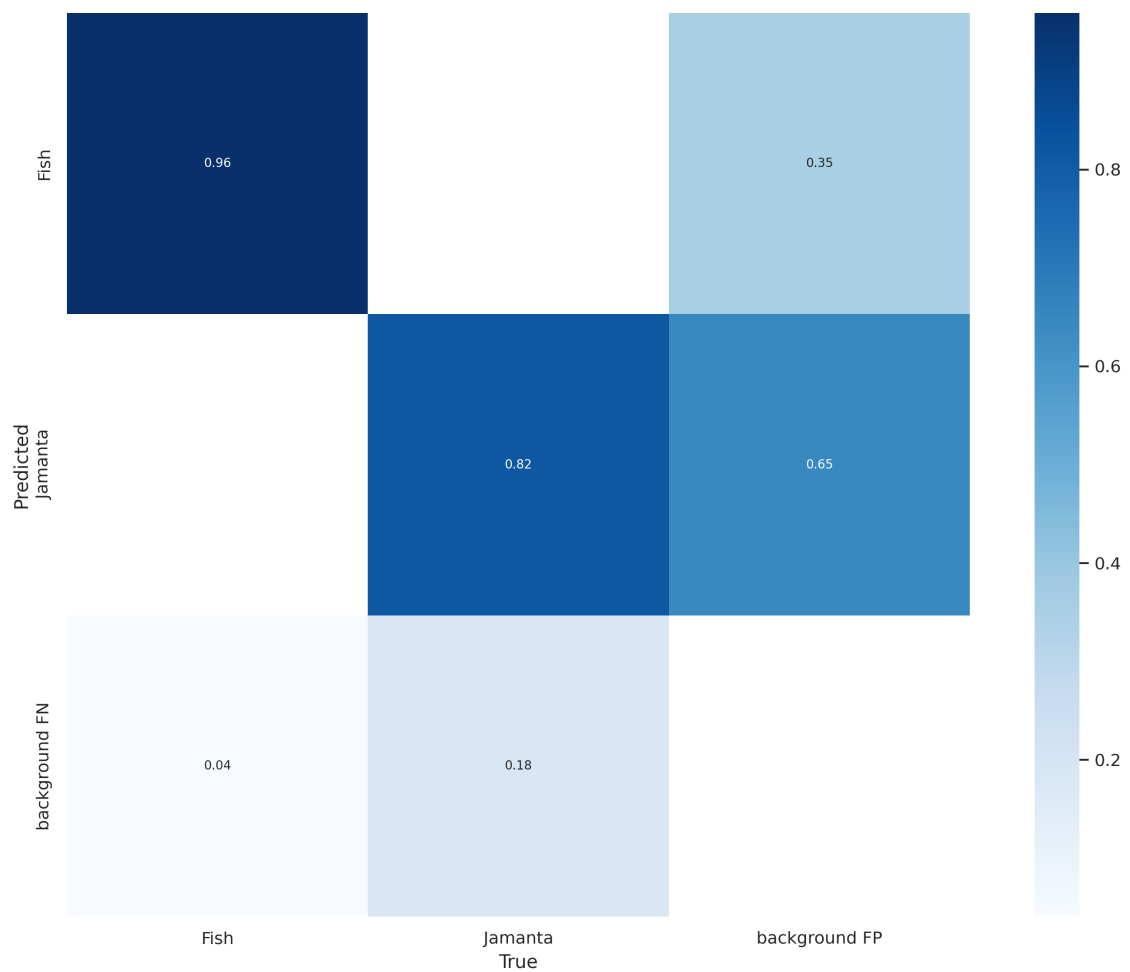


Figure 4.14: Confusion matrix for all the classes of the Jamanta's test data set

Similar to the approach for the Aquarium data set, the test for the Jamanta's data set should also be analyzed.

Analyzing the confusion matrix for the Jamanta's test set (Figure 4.14), the Fish class showed relevant changes, compared to the results from the validation set (Figure 4.12). For example, the precision for the Fish class of Jamanta's test set is larger than the precision for the same class in the validation set. This is a consequence of higher TP and lower FP values. As noted earlier in this section, as TP increases, the model has more confidence in the results and the precision also increases. On the other hand, there is a slight decrease in FN for the Fish class. Therefore, the model's confidence increases, and recall also increases.

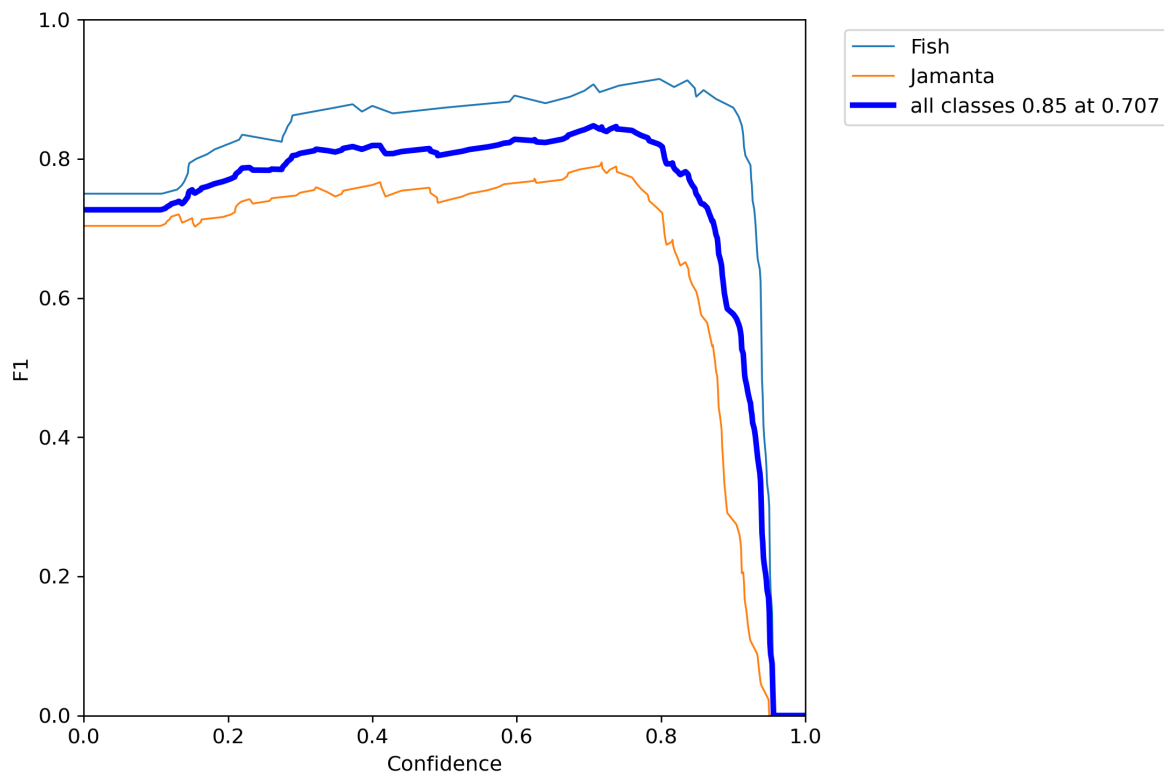


Figure 4.15: F1-curve for all the classes of the Jamanta's test data set

The effects of increasing precision and recall are shown in Figure 4.15, where the Fish class achieved an F1-score of over 80% at a confidence threshold of approximately 70%. For the Jamanta class, the model behaves better for the Jamanta's validation set, explained by the decrease in TP and the increase in FN, shown in Figure 4.14 from the test set. Following the analysis carried over the Fish class, the precision and recall decrease, meaning that the F1-score also decreases.

Nevertheless, both Jamanta's validation and test sets give similar results. Since the test set is used to confirm the results from the validation set, the fact that these results are similar shows that the model has the potential to generalize to other domains and be applied to data unknown by the model, making them more accurate as possible.

4.3 Underwater Image Processing

An experiment was carried out to evaluate the impact of applying two IP methods to the images of the Jamanta data set. Table 4.6 shows the results of this experiment when YOLOv7 is trained with the Reference data set, the Jamanta data set is processed with the Xiang et al. [77] method and the Jamanta data set processed with the Ghani & Isa [78] method. The following training sessions were conducted varying only the BS and LR hyperparameters since these are known to

have the most influence on the model’s performance. The standard deviations are obtained by computing the standard deviation of the entire list of mAP@0.5:0.95 values and the **E** column represents the epoch at which the mAP value was obtained.

Table 4.6: Percentage of the maximum mAP@0.5:0.95 values along with their respective standard deviations

		Reference		Xiang et al. [77]		Ghani & Isa [78]	
BS	LR	mAP@0.5:0.95 (%)	E	mAP@0.5:0.95 (%)	E	mAP@0.5:0.95 (%)	E
16	10^{-2}	72.26 ± 0.1	187	71.37 ± 0.1	142	70.44 ± 0.1	123
	10^{-3}	72.93 ± 0.1	75	72.99 ± 0.1	60	73.38 ± 0.1	100
	10^{-4}	72.62 ± 0.2	398	71.02 ± 0.2	253	70.24 ± 0.2	328
	10^{-5}	30.2 ± 0.09	397	34.3 ± 0.09	399	34.59 ± 0.1	387
32	10^{-2}	71.66 ± 0.1	114	68.89 ± 0.08	122	71.45 ± 0.09	98
	10^{-3}	73.89 ± 0.1	134	69.57 ± 0.1	86	72.34 ± 0.1	93
	10^{-4}	72.16 ± 0.2	313	69.14 ± 0.2	387	70.92 ± 0.1	371
	10^{-5}	35.61 ± 0.1	399	39.84 ± 0.1	398	38.76 ± 0.1	390
64	10^{-2}	69.06 ± 0.1	50	68.86 ± 0.09	134	70.21 ± 0.09	113
	10^{-3}	73.45 ± 0.2	122	69.63 ± 0.1	92	71.67 ± 0.1	93
	10^{-4}	75.9 ± 0.2	287	70.49 ± 0.17	205	71.83 ± 0.2	205
	10^{-5}	56.18 ± 0.2	397	61.44 ± 0.18	399	63.4 ± 0.2	398

When introducing the methods in Section 3.3, it was verified that the Ghani & Isa [78] is the most appropriate enhancement method to be applied on the Jamanta data set. Analyzing Table 4.6 shows that training with the Reference data set generally yields better results than training with enhanced data sets. Additionally, comparing both methods, the results for the Ghani & Isa [78] method are better than the results for the Xiang et al. [77] method. The difference in the results show that applying IP techniques can even worsen the model’s performance. Therefore, there is no guarantee that enhancing the images has a positive effect on the performance of the model. Although the images become more visually appealing, it does not imply that an OD model benefits from applying these techniques.

By analyzing the Reference and the Ghani & Isa columns in Table 4.6, the results are occasionally better for the Reference than for the Ghani & Isa. However, when the Ghani & Isa [78] method is applied to the Jamanta data set, it comes with a computational cost, which increases as the data set tends to increase. Since there are no significant differences between training YOLOv7 with the Reference and training with the data set processed with Ghani & Isa [78] method, there is no need to apply these techniques.

Finally, considering OD model apply filters while training, to extract the most important features, applying these filters on already processed images may lead to a loss of important

information. More specifically, both the Xiang et al. [77] and Ghani & Isa [78] methods, transform the histograms of the color channels to follow the Rayleigh distribution and apply filters to enhance the contours of the objects. This process is known to introduce extra noise [77], which is not ideal when the images are used to train an OD model. Consequently, the results for the Xiang et al. [77] method are worse, showing that IP techniques may have a negative influence on the performance of an OD model.

Chapter 5

Conclusions

In this study, a solution to efficiently analyze underwater videos is developed using YOLOv7 [36], known for its fast and accurate results. To achieve this, a dataset was created consisting of two classes, defined as the Jamanta dataset. The creation of this dataset involved analyzing videos with over 5 hours of duration, and manually annotating the marine animals present on the collected frames. Then, YOLOv7 was trained using the Jamanta dataset, and the mAP@0.5:0.95 ranged between 70% and 75%.

This study was proposed in collaboration with CEiiA, which held an internship between October 2022 and July 2023. The purpose of the internship was to apply OD models on underwater videos collected by telemetry equipment, defined as electronic tags. These tags are deployed in the Azores' region, enabling studies on the marine animals that inhabit the islands. The tags are placed around the marine animals for a maximum period of 8 hours. During this period, the tags record and collect important data, such as temperature and depth. Then, they are retrieved and the data is analyzed. The problem arises with the video analysis, which is time and staff-consuming, but may be solved by applying OD models. By developing a solution to efficiently detect the animals in underwater videos, researchers can save valuable time and resources. In addition, it enables them to connect recording data with quantitative and qualitative data and gather important information about the animals of the Azores' region.

When training YOLOv7, the concept of transfer learning was always considered, since this technique improves the training results. In practical terms, YOLOv7 is trained with a dataset, and the weights of this training session are used to train another dataset, preferably in a similar domain. The first experiments consisted of training YOLOv7 with the Jamanta dataset using pre-trained weights acquired from training YOLOv7 with the msCOCO [44] dataset. This dataset consists of 80 classes of common objects, which do not contain any class related to the underwater scene. Therefore, the first training sessions with the Jamanta dataset showed non-satisfactory results, even though pre-trained weights were used. To improve these results, another dataset was considered, consisting of seven classes of marine animals. This dataset was labeled as the Aquarium dataset and was used to train the YOLOv7 model. Then, the weights were used to train YOLOv7 with the Jamanta dataset and the results were significantly better.

In addition, combining the YOLOv7’s hyperparameters, such as BS, LR, WD, Mom. and IR was a crucial step to guarantee the best results and to decide the most appropriate hyperparameter combination. As mentioned in the literature [82], the BS and LR have the most impact on the model, especially with lower LR values.

When analyzing the results for the Jamanta dataset, it is clear that using the pre-trained weights from training YOLOv7 with the Aquarium dataset is an important step. Throughout all training sessions with the Jamanta dataset, the model reaches mAP@0.5:0.95 values near 70% for the validation set, with no signs of overfitting. For the test set, the model’s behavior was similar to that of the validation set. Consequently, the model shows good generalization capacity, which is crucial for future applications. However, the results of training the model with the Aquarium dataset were not as good, with mAP@0.5:0.95 varying between 45% and 50% for the validation set. For the test set, some of the classes showed difficulties in generalizing while other classes showed satisfactory results. Nevertheless, the focus of this study does not range from optimizing the results for training with the Aquarium dataset, since it was only considered to improve the model’s performance with the Jamanta dataset.

Throughout this study, several challenges and limitations were present. First and foremost, acquiring the data for the Jamanta dataset was a difficult task, since the images are affected by how light travels underwater, due to absorption and scattering phenomena. The combination of these phenomena, joined by other external factors degrades the quality of underwater images. For example, the images are dominated by the blue color since it is the last color to be absorbed by the water. The abundance of low-quality data influences the performance of OD models since they highly depend on large amounts of high-quality data. Since the images from the jamanta videos suffer from these issues, the dataset’s size is not large enough to guarantee that, in other domains, the model yields good results. To fix this issue, two IP methods were applied to the Jamanta dataset and the effects of these techniques on the performance of YOLOv7 were evaluated. By only varying the BS and LR, it was verified that enhancing the images did not have a positive influence on training results. In fact, applying these methods may even worsen the results, which was also mentioned in the literature [6, 19]. However, the chosen methods do not imply that all IP techniques perform poorly, since these experiments are empirical, and may vary from method to method.

Additionally, other underwater datasets could be used to pre-train YOLOv7, such as the Fish4Knowledge [84]. However, the Aquarium dataset had already been used by CEiiA, with good results. In addition, YOLOv7 was the chosen model since it is well known that the YOLO series achieves fast and accurate results, which is CEiiA’s purpose for this project. Furthermore, the chosen hyperparameter optimization method served as a way to select the most appropriate hyperparameters, while reducing the number of combinations needed. Despite obtaining good results, other optimization approaches are available. For example, YOLOv7 allows the hyperparameters to be optimized for a pre-defined epoch number. Then, the model outputs the hyperparameter file with the optimized hyperparameter combination. In addition, grid search and random search are also alternatives to optimize the hyperparameters.

Moreover, after the development of the Jamanta dataset, a second dataset was developed and defined as the Whale Shark dataset. The images collected had higher quality than the images from the Jamanta dataset since a different camera was used for the whale sharks' videos. Three classes were identified: Whale Shark, Fish, and Remora. Due to a lack of time, it was not possible to train YOLOv7 with this dataset. It was also preferred that the focus should be on improving the results from the Jamanta dataset.

The work developed within this thesis is suitable to be used in a real-world application but the work does not remain stationary. More specifically, the analyses conducted for both the Aquarium and Jamanta datasets should be applied to the Whale Shark dataset. Then, the Jamanta and Whale Shark datasets should be combined to guarantee that the model's generalization capacity improves even further. Furthermore, the effects of other IP techniques on the performance of OD models can be assessed. However, there is the need to theoretically prove that IP techniques have a strictly negative influence on the performance of the model, disregarding the empirical proofs shown in this study and other studies. This way, it is guaranteed that applying IP techniques as a pre-processing step should be avoided. Additionally, when selecting the best combination of hyperparameters, it was verified that WD has the least influence during training. Even though several factors may influence this occurrence, the effects of WD should still be studied. Finally, other OD models may be used. For example, two-stage object detectors, such as the R-CNN series [25–27], are known to achieve better results than the YOLO series, even if training lasts longer. This way, it is possible to understand the differences between the models and decide which is the most appropriate model for the datasets developed during this study.

In conclusion, the work developed in this study provides two custom underwater datasets ready to be used for numerous CV applications, not being restricted to the OD field. More importantly, this study enables the efficient and accurate analysis of underwater videos, which aids researchers in optimizing their research studies. In addition, the application of IP techniques and the study of their effects on the performance of an OD model is also useful for several reasons. First, this analysis guarantees that there is no need to pre-process the data with IP techniques. Second, the application of IP techniques implies an increase in computational cost as the dataset increases. Considering the model's results are similar with and without these techniques, their application is not feasible. Third, even though IP techniques make the images more visually appealing for humans, it does not mean that the same occurs for OD models.

Overall, the objectives of this thesis were achieved and it is intended that the methodology followed during this study serves as a guide for future work developed in the CV field, more specifically related to the underwater medium.

Appendix A

Results of Training YOLOv7

A.1 Aquarium dataset

A.1.1 Test dataset

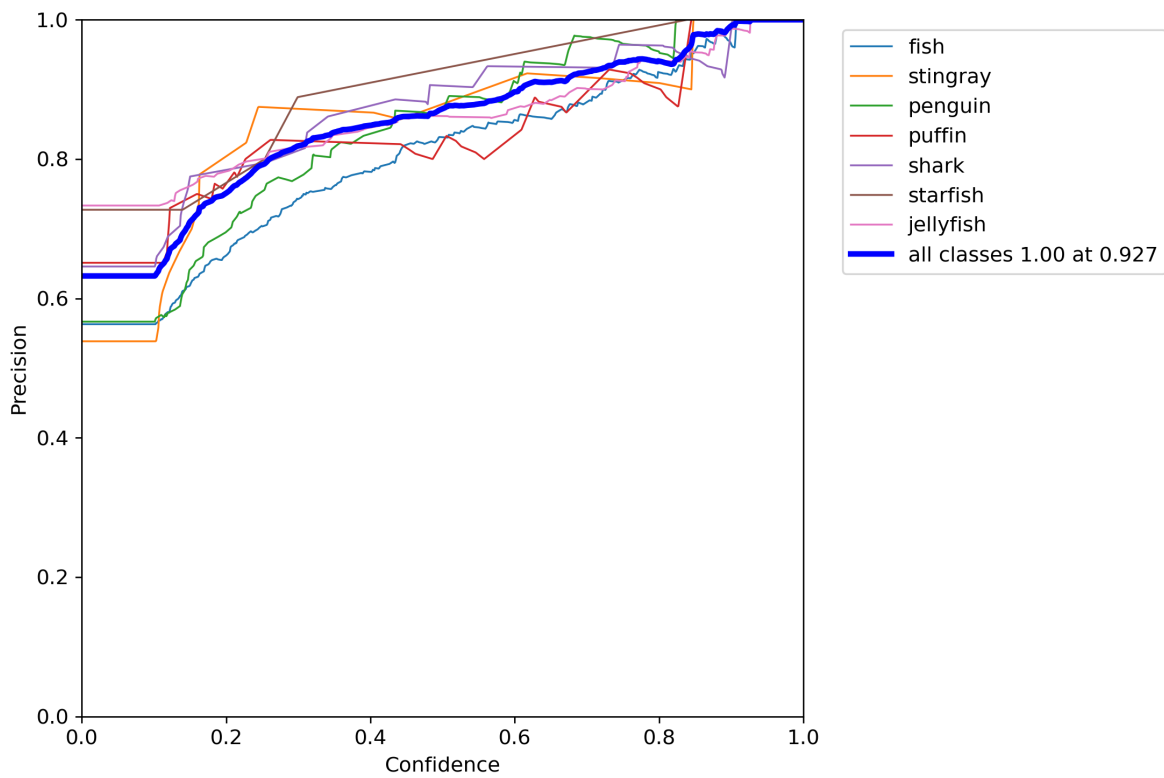


Figure A.1: Plot for the Precision curve measured over increasing confidence thresholds for the Aquarium's test data set

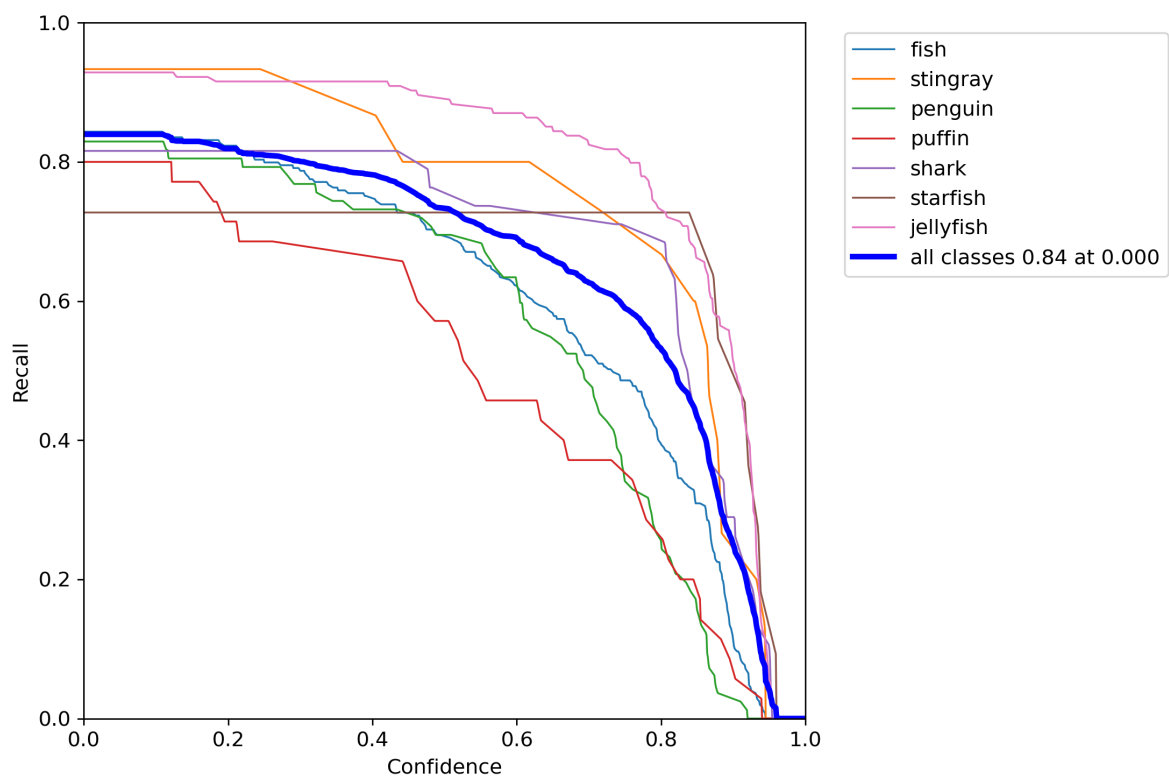
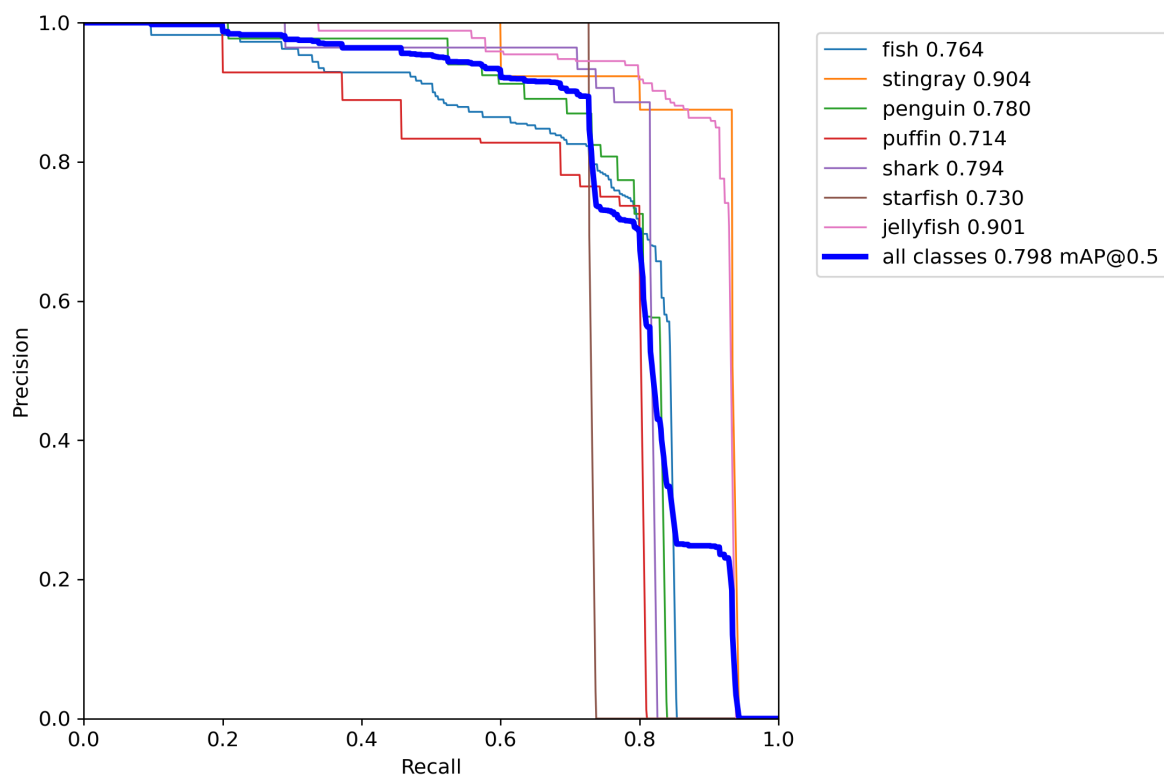
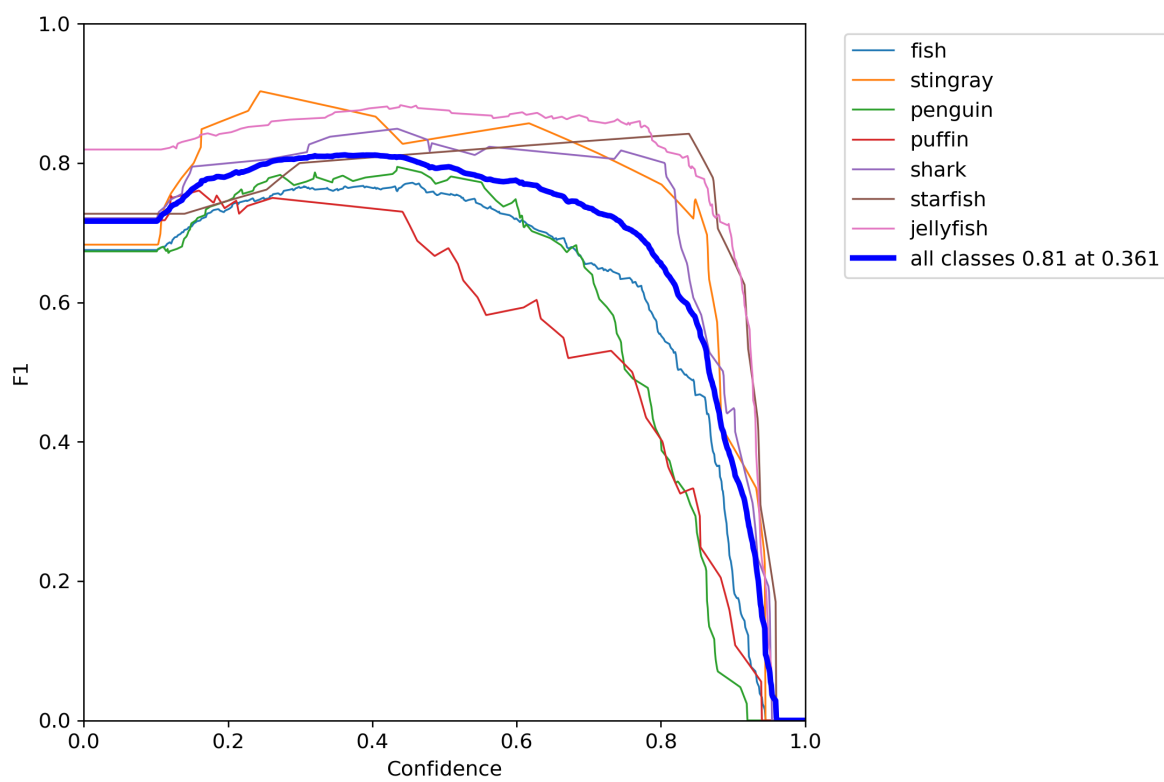


Figure A.2: Plot for the Recall curve measured over increasing confidence thresholds for the Aquarium's test data set



(a) Precision-Recall curve

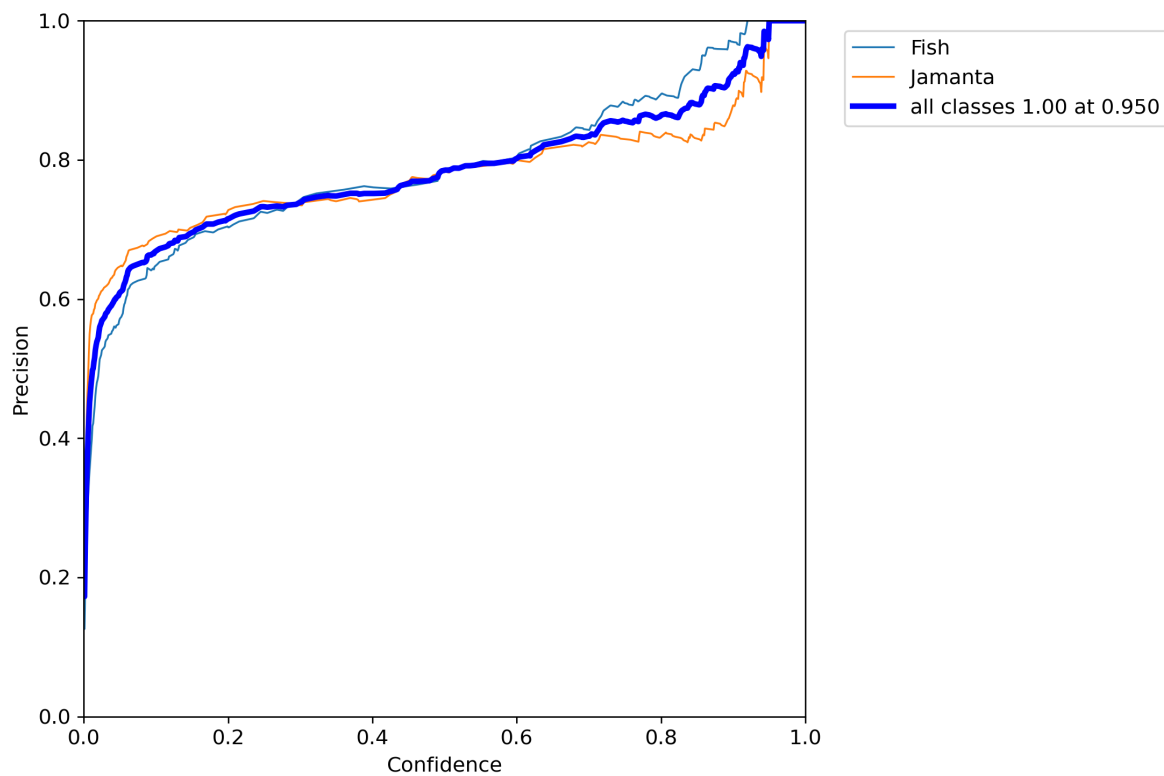


(b) F1-score curve over increasing confidence thresholds

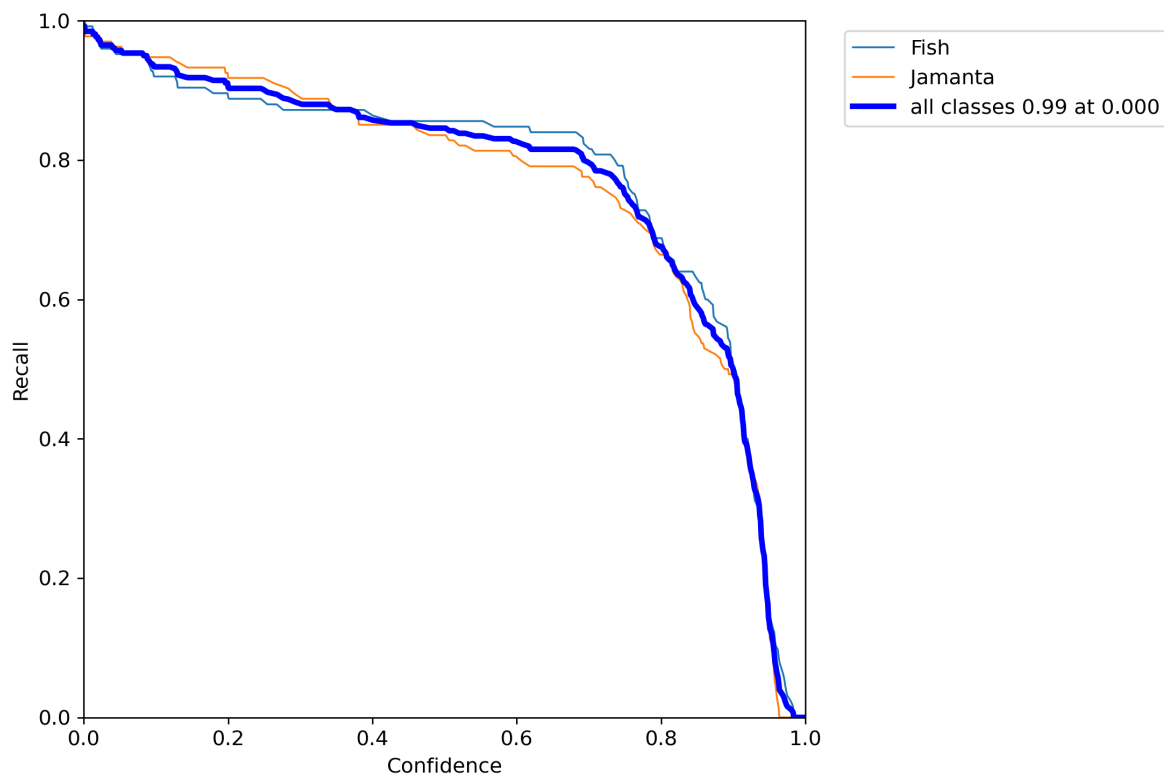
Figure A.3: Plots for the Precision-Recall (a) and F1-score (b) curves for the Aquarium's test dataset

A.2 Jamanta dataset

A.2.1 Validation dataset

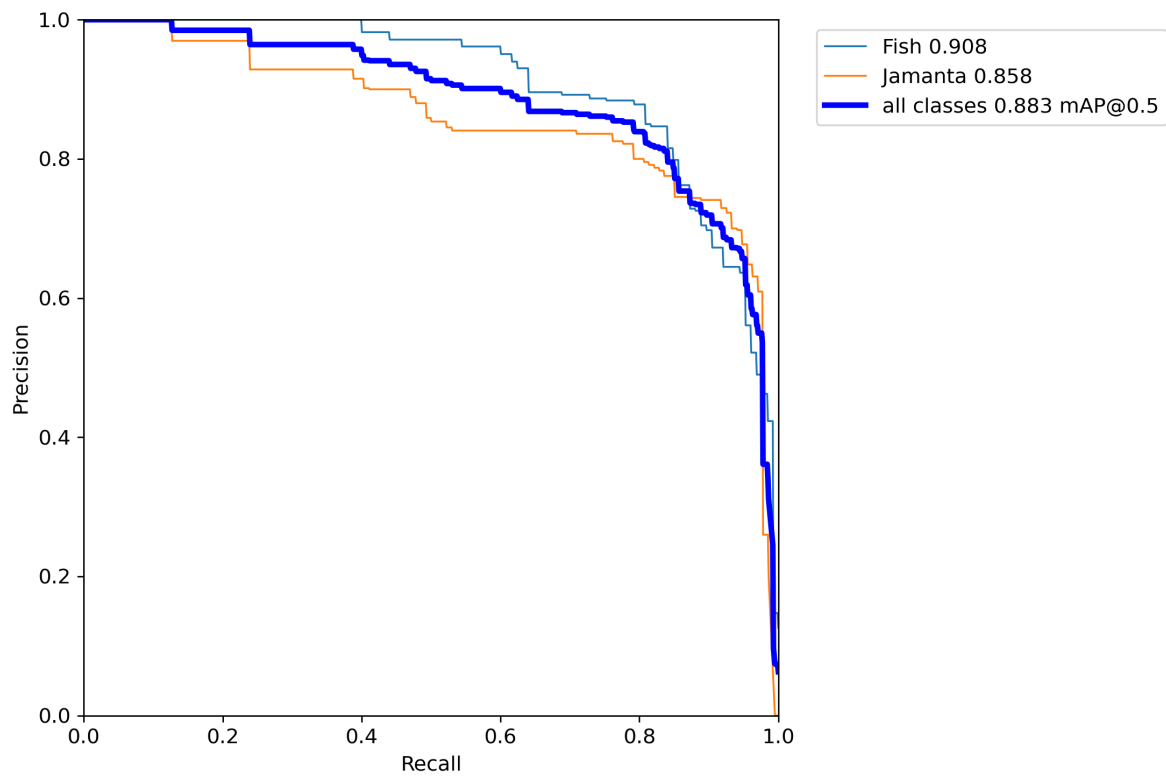


(a) Precision curve

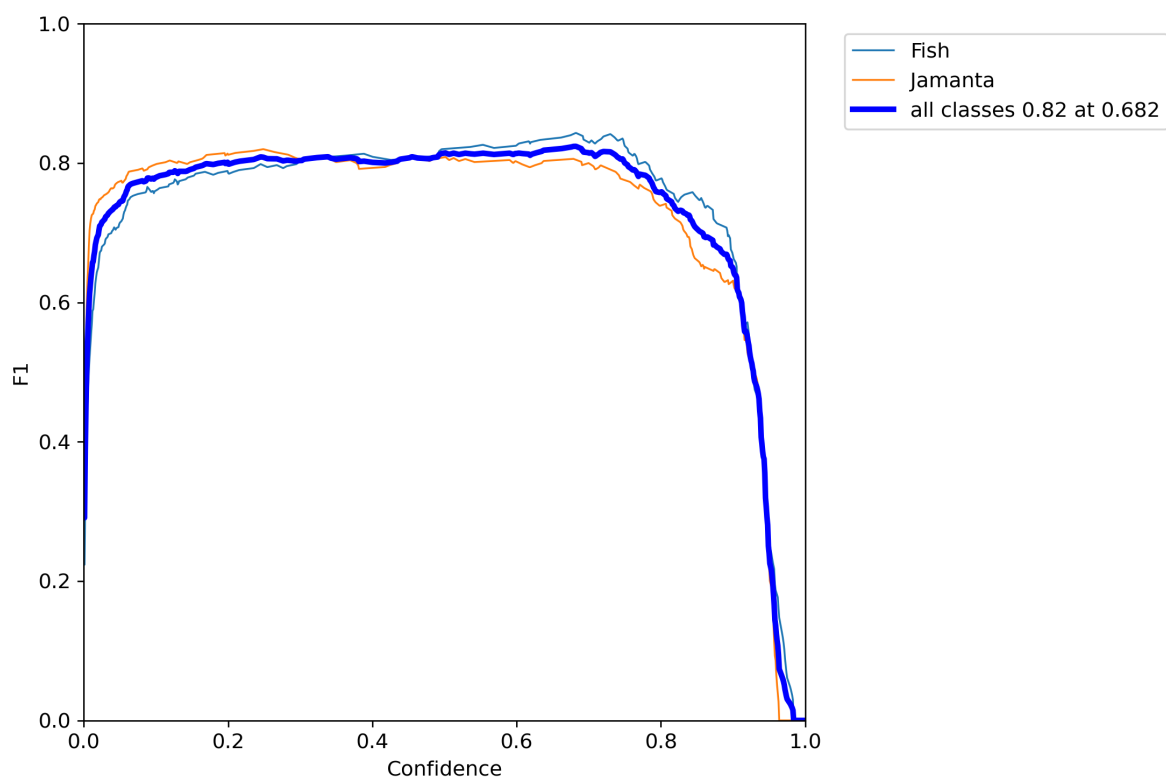


(b) Recall curve

Figure A.4: Plots for the Precision (a) and Recall (b) Recall curve measured over increasing confidence thresholds for the Jamanta's validation dataset



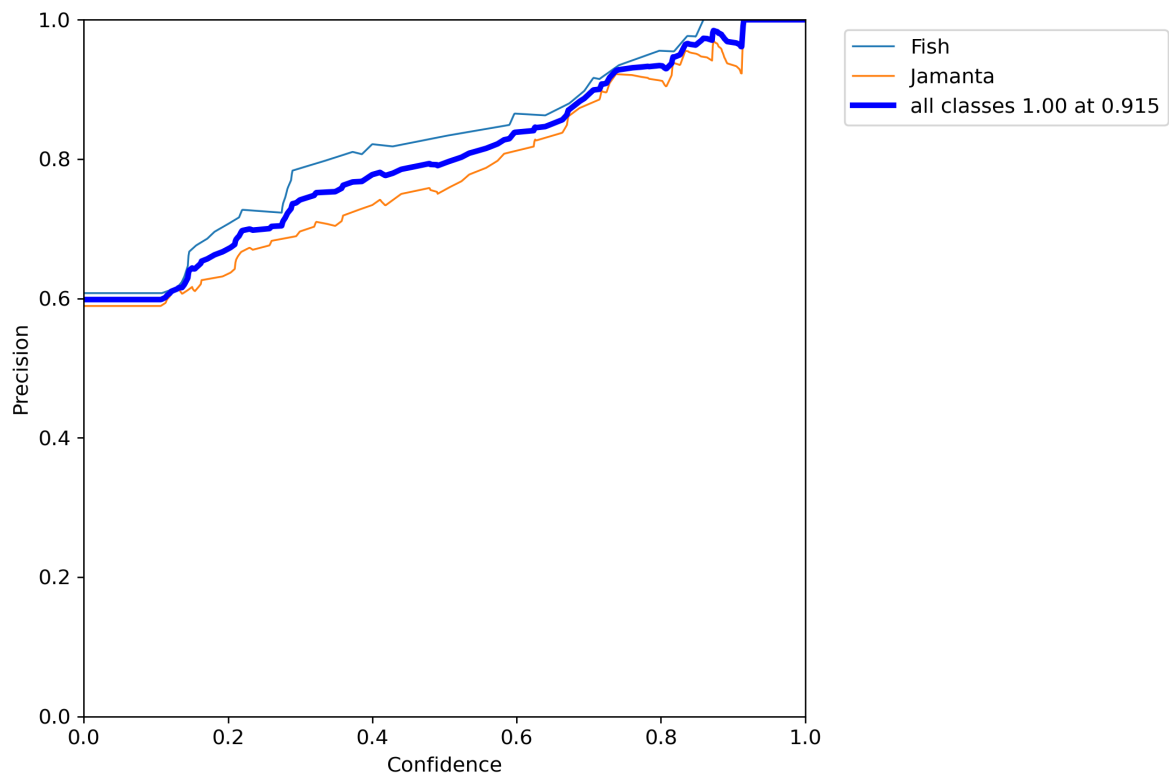
(a) Precision-Recall curve



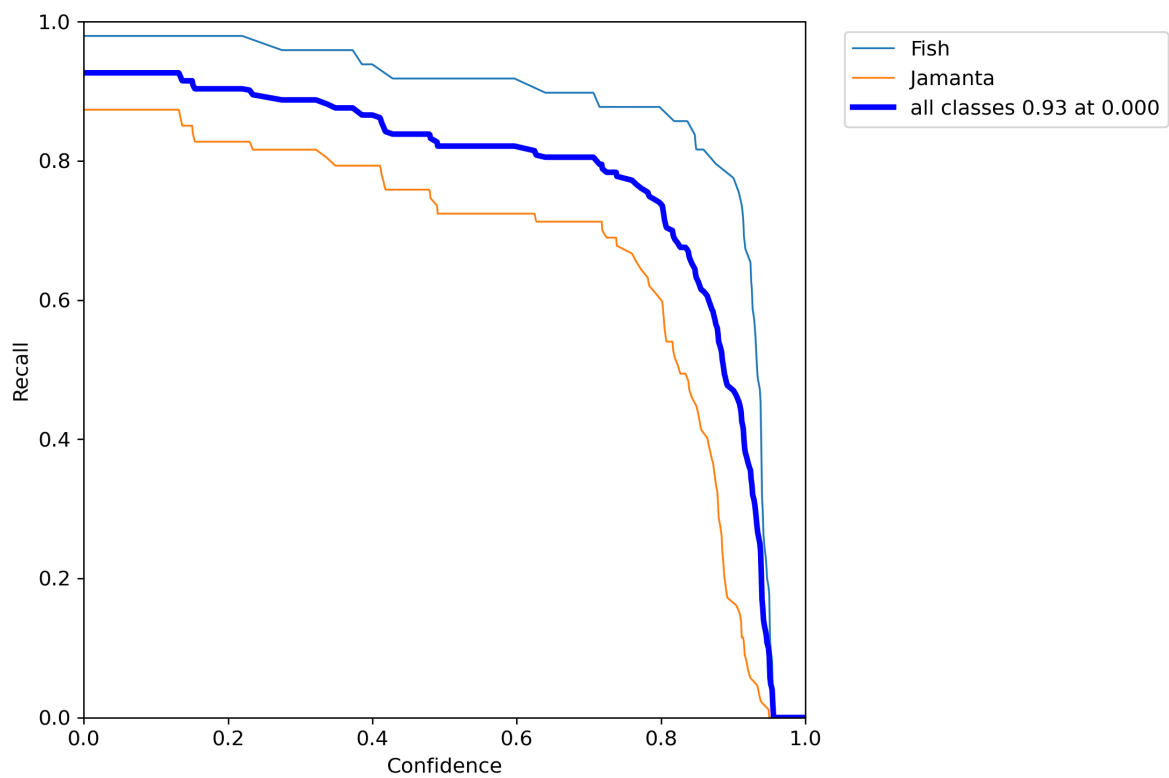
(b) F1-score curve

Figure A.5: Plots for the Precision-Recall (a) and F1-score (b) curves for the Jamanta's validation dataset

A.2.2 Test set



(a) Precision curve



(b) Recall curve

Figure A.6: Plots for the Precision (a) and Recall (b) Recall curve measured over increasing confidence thresholds for the Jamanta's test dataset

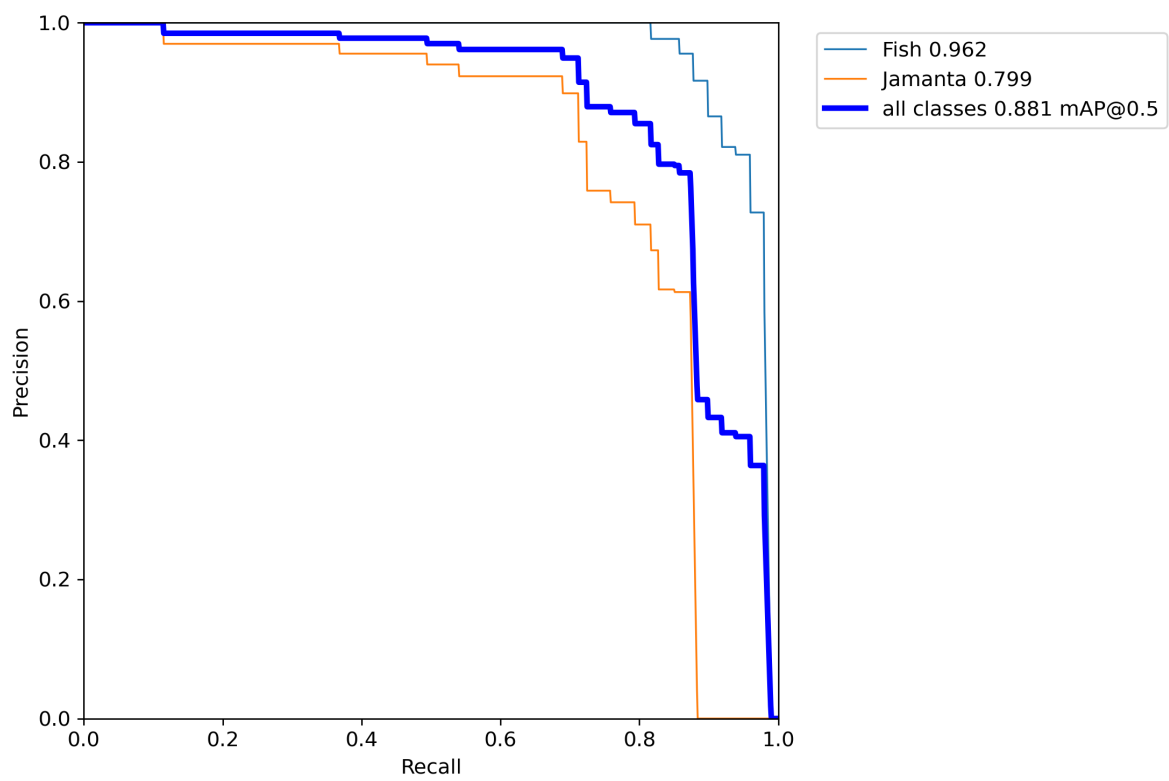
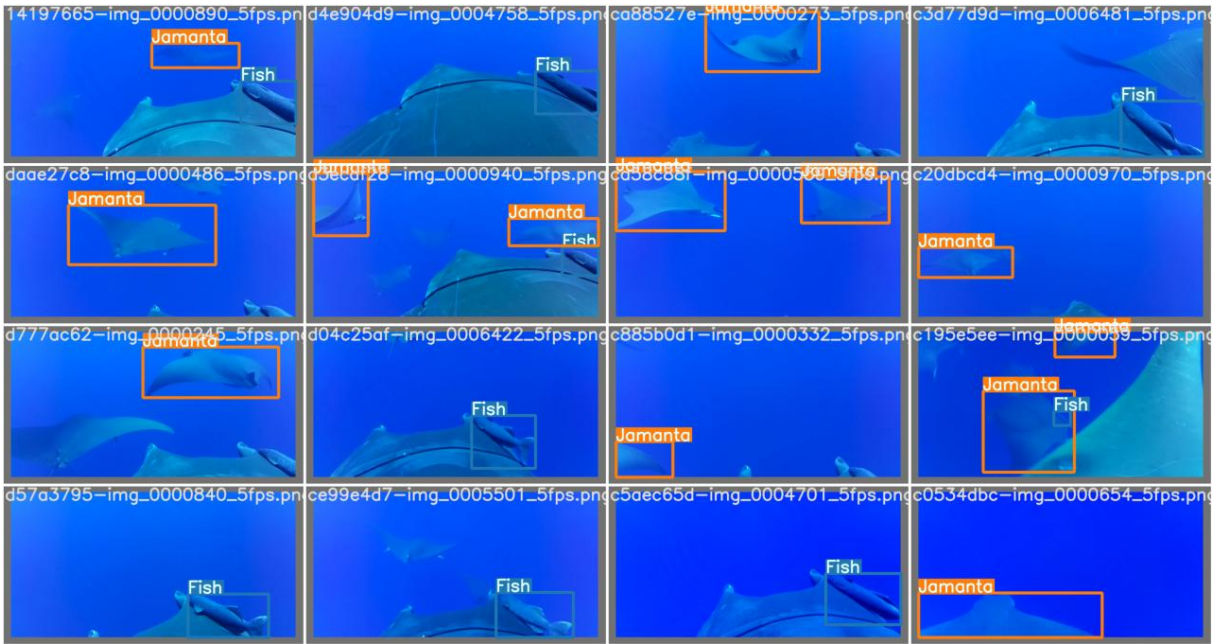
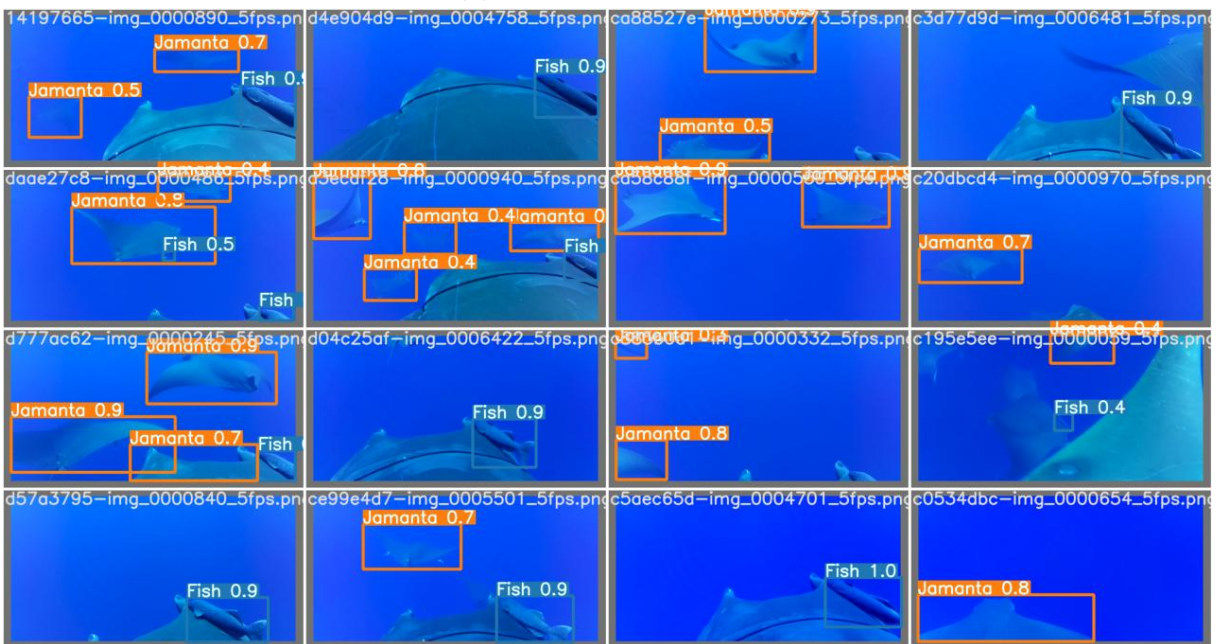


Figure A.7: Plot for the Precision-Recall curve for the Jamanta's test dataset of the Jamanta's test dataset



(a) Ground-truth labels



(b) Model's predictions

Figure A.8: Ground-truth labels (a) and model's predictions for the Jamanta's test dataset

Bibliography

- [1] J. Whitfield Gibbons and Kimberly M. Andrews. PIT Tagging: Simple Technology at Its Best. *BioScience*, 54(5):447, 2004.
- [2] Nigel E. Hussey, Steven T. Kessel, Kim Aarestrup, Steven J. Cooke, Paul D. Cowley, Aaron T. Fisk, Robert G. Harcourt, Kim N. Holland, Sara J. Iverson, John F. Kocik, Joanna E. Mills Flemming, and Fred G. Whoriskey. Aquatic animal telemetry: A panoramic window into the underwater world. *Science*, 348(6240):1255642, 2015.
- [3] Eric Hockersmith and John Beeman. *A History of Telemetry in Fishery Research*, pages 7–19. 01 2012.
- [4] Christian Rutz and Graeme C. Hays. New frontiers in biologging science. *Biology Letters*, 5(3):289–292, June 2009.
- [5] Eva B. Thorstad, Audun H. Rikardsen, Ahmet Alp, and Finn Okland. The use of electronic tags in fish research – an overview of fish telemetry methods. *Turkish Journal of Fisheries and Aquatic Sciences*, 13(5), 2014.
- [6] Shubo Xu, Minghua Zhang, Wei Song, Haibin Mei, Qi He, and Antonio Liotta. A Systematic Review and Analysis of Deep Learning-based Underwater Object Detection. *Neurocomputing*, page S0925231223000656, January 2023.
- [7] Sheezan Fayaz, Shabir A. Parah, and G. J. Qureshi. Underwater object detection: architectures and algorithms – a comprehensive review. *Multimedia Tools and Applications*, 81(15):20871–20916, June 2022.
- [8] Salma P. González-Sabbagh and Antonio Robles-Kelly. A Survey on Underwater Computer Vision. *ACM Computing Surveys*, page 3578516, January 2023.
- [9] Mahmoud Hassaballah and Ali Ismail Awad. *Deep Learning in Computer Vision: Principles and Applications*. CRC Press, First edition. | Boca Raton, FL : CRC Press/Taylor and Francis, 2020. |, 1 edition, March 2020.
- [10] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee. A survey of modern deep learning based object detection models. *Digital Signal Processing*, 126:103514, June 2022.

-
- [11] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep Learning vs. Traditional Computer Vision. In Kohei Arai and Supriya Kapoor, editors, *Advances in Computer Vision*, volume 943, pages 128–144. Springer International Publishing, Cham, 2020. Series Title: Advances in Intelligent Systems and Computing.
- [12] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object Detection in 20 Years: A Survey. January 2023. arXiv:1905.05055 [cs].
- [13] Anamika Dhillon and Gyanendra K. Verma. Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence*, 9(2):85–112, June 2020.
- [14] Tausif Diwan, G. Anirudh, and Jitendra V. Tembhurne. Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*, 82(6):9243–9275, March 2023.
- [15] Mostafa S. Ibrahim, Amr A. Badr, Mostafa R. Abdallah, and Ibrahim F. Eissa. Bounding Box Object Localization Based On Image Superpixelization. *Procedia Computer Science*, 13:108–119, 2012.
- [16] T S Huang. Computer Vision: Evolution and Promise.
- [17] Qi Chen, Anumol Mathai, Xiping Xu, and Xin Wang. A Study into the Effects of Factors Influencing an Underwater, Single-Pixel Imaging System's Performance. *Photonics*, 6(4):123, November 2019.
- [18] Jingchun Zhou, Xiaojing Wei, Jinyu Shi, Weishen Chu, and Weishi Zhang. Underwater image enhancement method with light scattering characteristics. *Computers and Electrical Engineering*, 100:107898, May 2022.
- [19] Jiashuo Zhang, Linlin Zhu, Liheng Xu, and Qian Xie. Research on the Correlation between Image Enhancement and Underwater Object Detection. In *2020 Chinese Automation Congress (CAC)*, pages 5928–5933, Shanghai, China, November 2020. IEEE.
- [20] Youzi Xiao, Zhiqiang Tian, Jiachen Yu, Yinshu Zhang, Shuai Liu, Shaoyi Du, and Xuguang Lan. A review of object detection based on deep learning. *Multimedia Tools and Applications*, 79(33-34):23729–23791, September 2020.
- [21] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518, Kauai, HI, USA, 2001. IEEE Comput. Soc.
- [22] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893, San Diego, CA, USA, 2005. IEEE.

-
- [23] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008. IEEE.
- [24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May 2017.
- [25] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, October 2014. arXiv:1311.2524 [cs].
- [26] Ross Girshick. Fast R-CNN, September 2015. arXiv:1504.08083 [cs].
- [27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, January 2016. arXiv:1506.01497 [cs].
- [28] Thorsten Hoeser and Claudia Kuenzer. Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review-Part I: Evolution and Recent Trends. *Remote Sensing*, 12(10):1667, May 2020.
- [29] Petru Soviany and Radu Tudor Ionescu. Optimizing the Trade-Off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction. In *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 209–214, Timisoara, Romania, September 2018. IEEE.
- [30] Shubo Xu, Minghua Zhang, Wei Song, Haibin Mei, Qi He, and Antonio Liotta. A systematic review and analysis of deep learning-based underwater object detection. *Neurocomputing*, 527:204–232, March 2023.
- [31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. May 2016. arXiv:1506.02640 [cs].
- [32] Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger. December 2016. arXiv:1612.08242 [cs].
- [33] Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement. April 2018. arXiv:1804.02767 [cs].
- [34] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, April 2020. arXiv:2004.10934 [cs, eess].
- [35] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications, September 2022. arXiv:2209.02976 [cs].

-
- [36] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, July 2022. arXiv:2207.02696 [cs].
- [37] Yuhao Lai, Ruijun Ma, Yu Chen, Tao Wan, Rui Jiao, and Huandong He. A Pineapple Target Detection Method in a Field Environment Based on Improved YOLOv7. *Applied Sciences*, 13(4):2691, February 2023.
- [38] Md. Moniruzzaman, Syed Mohammed Shamsul Islam, Mohammed Bennamoun, and Paul Lavery. Deep Learning on Underwater Marine Object Detection: A Survey. In Jacques Blanc-Talon, Rudi Penne, Wilfried Philips, Dan Popescu, and Paul Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 10617, pages 150–160. Springer International Publishing, Cham, 2017. Series Title: Lecture Notes in Computer Science.
- [39] Jialun Dai, Ruchen Wang, Haiyong Zheng, Guangrong Ji, and Xiaoyan Qiao. ZooplanktoNet: Deep convolutional network for zooplankton classification. In *OCEANS 2016 - Shanghai*, pages 1–6, Shanghai, China, April 2016. IEEE.
- [40] Hansang Lee, Minseok Park, and Junmo Kim. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3713–3717, Phoenix, AZ, USA, September 2016. IEEE.
- [41] Yanfeng Gong, Jun Luo, Hongliang Shao, and Zhixue Li. A transfer learning object detection model for defects detection in X-ray images of spacecraft composite structures. *Composite Structures*, 284:115136, March 2022.
- [42] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.
- [43] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, June 2009. IEEE.
- [44] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context. February 2015. arXiv:1405.0312 [cs].
- [45] Osva Antonio Montesinos López, Abelardo Montesinos López, and José Crossa. *Multivariate Statistical Machine Learning Methods for Genomic Prediction*. Springer International Publishing, Cham, 2022.
- [46] Hao Gu, Yu Wang, Sheng Hong, and Guan Gui. Blind Channel Identification Aided Generalized Automatic Modulation Recognition Based on Deep Learning. *IEEE Access*, 7:110722–110729, 2019.

- [47] Wang Zhiqiang and Liu Jun. A review of object detection based on convolutional neural network. In *2017 36th Chinese Control Conference (CCC)*, pages 11104–11109, Dalian, China, July 2017. IEEE.
- [48] Joao Nuno Tavares Dm-Fcup. PART 1. NEURAL NETWORKS. 2023.
- [49] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. November 2018. arXiv:1811.03378 [cs].
- [50] Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines.
- [51] Muhamad Yani, M.T. S Budhi Irawan, Si., and S.T. Casi Setiningsih, M.T. Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry’s Nail. *Journal of Physics: Conference Series*, 1201(1):012052, May 2019.
- [52] Md. Janibul Alam Soeb, Md. Fahad Jubayer, Tahmina Akanjee Tarin, Muhammad Rashed Al Mamun, Fahim Mahafuz Ruhad, Aney Parven, Nabisab Mujawar Mubarak, Soni Lanka Karri, and Islam Md. Meftaul. Tea leaf disease detection and identification based on YOLOv7 (YOLO-T). *Scientific Reports*, 13(1):6078, April 2023.
- [53] Rafael Padilla, Sergio L. Netto, and Eduardo A. B. da Silva. A Survey on Performance Metrics for Object-Detection Algorithms. In *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 237–242, Niterói, Brazil, July 2020. IEEE.
- [54] Rafael Padilla, Wesley L. Passos, Thadeu L. B. Dias, Sergio L. Netto, and Eduardo A. B. da Silva. A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. *Electronics*, 10(3):279, January 2021.
- [55] Sangwoo Seo, Youngmin Kim, Hyo-Jeong Han, Woo Chan Son, Zhen-Yu Hong, Insuk Sohn, Jooyong Shim, and Changha Hwang. Predicting Successes and Failures of Clinical Trials With Outer Product-Based Convolutional Neural Network. *Frontiers in Pharmacology*, 12:670670, June 2021.
- [56] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions, September 2014. arXiv:1409.4842 [cs].
- [57] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, April 2015. arXiv:1409.1556 [cs].
- [58] Sara Bouraya and Abdessamad Belangour. Deep Learning based Neck Models for Object Detection: A Review and a Benchmarking Study. *International Journal of Advanced Computer Science and Applications*, 12(11), 2021.

-
- [59] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning Non-maximum Suppression. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6469–6477, Honolulu, HI, July 2017. IEEE.
- [60] Jincheng Chen, Shoujun Bai, Guoyang Wan, and Yunfei Li. Research on YOLOv7-based defect detection method for automotive running lights. *Systems Science & Control Engineering*, 11(1):2185916, December 2023.
- [61] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, March 2015. arXiv:1502.03167 [cs].
- [62] Margrit Kasper-Eulaers, Nico Hahn, Stian Berger, Tom Sebulonsen, Øystein Myrland, and Per Egil Kummervold. Short Communication: Detecting Heavy Goods Vehicles in Rest Areas in Winter Conditions Using YOLOv5. *Algorithms*, 14(4):114, March 2021.
- [63] Duane Edgington and Danelle Cline. Hyperparameter Evolution For Deepsea Detection and Tracking.
- [64] Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, November 2020.
- [65] Tomoumi Takase. Dynamic batch size tuning based on stopping criterion for neural network training. *Neurocomputing*, 429:1–11, March 2021.
- [66] Gyoung S. Na. Efficient learning rate adaptation based on hierarchical optimization approach. *Neural Networks*, 150:326–335, June 2022.
- [67] Ismoilov Nusrat and Sung-Bong Jang. A Comparison of Regularization Techniques in Deep Neural Networks. *Symmetry*, 10(11):648, November 2018.
- [68] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, January 1999.
- [69] Smitha Raveendran, Mukesh D. Patil, and Gajanan K. Birajdar. Underwater image enhancement: a comprehensive review, recent trends, challenges and applications. *Artificial Intelligence Review*, 54(7):5413–5467, October 2021.
- [70] Kai Hu, Chenghang Weng, Yanwen Zhang, Junlan Jin, and Qingfeng Xia. An overview of underwater vision enhancement: From traditional methods to recent deep learning. *Journal of Marine Science and Engineering*, 10(2):241, 2022.
- [71] Marco Reggiannini and Davide Moroni. The Use of Saliency in Underwater Computer Vision: A Review. *Remote Sensing*, 13(1):22, December 2020.
- [72] Miao Yang, Jintong Hu, Chongyi Li, Gustavo Rohde, Yixiang Du, and Ke Hu. An In-Depth Survey of Underwater Image Enhancement and Restoration. *IEEE Access*, 7:123638–123657, 2019.

- [73] Donna M Kocak, Fraser R Dalglish, Frank M Caimi, and Yoav Y Schechner. A focus on recent developments and trends in underwater imaging. *Marine Technology Society Journal*, 42(1):52, 2008.
- [74] Jacob W. Brownscombe, Elodie J. I. Lédée, Graham D. Raby, Daniel P. Struthers, Lee F. G. Gutowsky, Vivian M. Nguyen, Nathan Young, Michael J. W. Stokesbury, Christopher M. Holbrook, Travis O. Brenden, Christopher S. Vandergoot, Karen J. Murchie, Kim Whoriskey, Joanna Mills Flemming, Steven T. Kessel, Charles C. Krueger, and Steven J. Cooke. Conducting and interpreting fish telemetry studies: considerations for researchers and resource managers. *Reviews in Fish Biology and Fisheries*, 29(2):369–400, June 2019.
- [75] Jd Stewart, T Smith, G Marshall, K Abernathy, Ia Fonseca-Ponce, N Froman, and Gmw Stevens. Novel applications of animal-borne Crittercams reveal thermocline feeding in two species of manta ray. *Marine Ecology Progress Series*, 632:145–158, December 2019.
- [76] Manon Kok, Jeroen D. Hol, and Thomas B. Schön. Using Inertial Sensors for Position and Orientation Estimation. *Foundations and Trends® in Signal Processing*, 11(1-2):1–153, 2017. arXiv:1704.06053 [cs].
- [77] Wending Xiang, Ping Yang, Shuai Wang, Bing Xu, Hui Liu, 1.Key Laboratory of Adaptive Optics, Chinese Academy of Sciences, Chengdu 610209, China, 2.Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu 610209, China, 3.University of Chinese Academy of Sciences, Beijing 100049, China, and 4.Yantai Institute of Coastal Zone Research, Chinese Academy of Sciences, Yantai 264003, China. Underwater image enhancement based on red channel weighted compensation and gamma correction model. *Opto-Electronic Advances*, 1(10):18002401–18002409, 2018.
- [78] Ahmad Shahrizan Abdul Ghani and Nor Ashidi Mat Isa. Underwater image quality enhancement through integrated color model with Rayleigh distribution. *Applied Soft Computing*, 27:219–230, February 2015.
- [79] Kashif Iqbal, Rosalina Abdul Salam, Azam Osman, and Abdullah Zawawi Talib. Underwater Image Enhancement Using an Integrated Colour Model. 2007.
- [80] Kashif Iqbal, Michael Odetayo, Anne James, Rosalina Abdul Salam, and Abdullah Zawawi Hj Talib. Enhancing the low quality images using Unsupervised Colour Correction Method. In *2010 IEEE International Conference on Systems, Man and Cybernetics*, pages 1703–1709, Istanbul, Turkey, October 2010. IEEE.
- [81] Ahmad Shahrizan Abdul Ghani. Underwater image quality enhancement through Rayleigh-stretching and averaging image planes. 2014.
- [82] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay, April 2018. arXiv:1803.09820 [cs, stat].

-
- [83] S. Razavi and B. A. Tolson. A New Formulation for Feedforward Neural Networks. *IEEE Transactions on Neural Networks*, 22(10):1588–1598, October 2011.
- [84] Robert B. Fisher, Yun-Heh Chen-Burger, Daniela Giordano, Lynda Hardman, and Fang-Pang Lin, editors. *Fish4Knowledge: Collecting and Analyzing Massive Coral Reef Fish Video Data*, volume 104 of *Intelligent Systems Reference Library*. Springer International Publishing, Cham, 2016.