FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Artificial Intelligence for Automated Marine Growth Classification

João Afonso Borges Carvalho

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Andry Maykol Gomes Pinto

July 24, 2023

Resumo

A inspeção de estruturas marítimas é dificultada pela acumulação de crescimento marinho nas estruturas. O crescimento marinho afeta a estabilidade e integridade das estruturas, ao mesmo tempo em que impede a inspeção adequada da estrutura. Em consequência, as empresas precisam contratar especialistas que avaliam manualmente cada parte afetada da estrutura e agendam a manutenção onde é mais necessária. Ambientes adversos subaquáticos tornam difícil a tarefa de capturar e analisar imagens subaquáticas da estrutura, pois requer veículos especializados como ROVs para realizar essas operações e porque os ambientes subaquáticos impactam diretamente a qualidade das imagens. Este trabalho propõe utilizar algoritmos modernos de aprendizagem computacional para efetuar segmentação de imagem com o intuito de identificar regiões de crescimento marinho em imagens subaquáticas. Isto permitirá reduzir a carga de trabalho manual necessária para calendarizar processos de manutenção e aumentar o grau de automatização deste processo. Além disso, é proposto um algoritmo que gera novas imagens a partir de recortes localizados nas imagens originais como solução para ultrapassar a dificuldade de treinar algoritmos de aprendizagem computacional num dataset de dimensões reduzidas.

Abstract

Offshore structure inspection is obstructed by marine growth accumulation in the structure. Marine growth impacts the stability and integrity of offshore structures, while simultaneously preventing inspection of the structure. In consequence companies need to employ specialists that manually access each impacted part of the structure and schedule maintenance where it is most needed. Adverse subsea environments make the task of capturing and analysing underwater images of the structure difficult because it requires specialized vehicles, like Remotely Operated Vehicles, to perform these operations and because subsea environments directly impact the quality of the images. This work proposes to leverage state-of-the-art learning-based algorithms to perform image segmentation in order to identify regions of marine growth within underwater images. This will allow a reduction in the manual labour necessary to schedule maintenance processes for the structure and an increase in the degree of automation of the process. In addition, this work proposes an algorithm that generates new images by performing localized crops in the original data to overcome the challenges of training learning models in a small-scale dataset.

Acknowledgements

I would like to express my heartfelt gratitude to all those who have contributed to the completion of this thesis.

First and foremost, I am immensely grateful to my supervisor, Prof. Andry Maykol Gomes Pinto, for their invaluable guidance, and support throughout this research journey. Their expertise, patience, and unwavering commitment to academic excellence have been instrumental in shaping this thesis. A special thanks is also directed at Pedro Leite, I am extremely grateful for their support and guidance throughout this thesis, which proved invaluable to the final product.

I would like to express gratitude to the guys at Ocean Infinity, Pedro Costa, José and Filipe for their contributions and expertise to this thesis.

Furthermore I would like to extend heartfelt gratitude to all my family for their constant support, my friends in FakeFeup who were part of my daily life for the last 5 years and Ana for her unwavering support throughout this whole process.

João Carvalho

"You should be glad that bridge fell down.
I was planning to build thirteen more to that same design"

Isambard Kingdom Brunel

Contents

1 Introduction **1**
- 1.1 Context and Motivation . . . 1
- 1.2 Objectives . . . 3
- 1.3 Work structure . . . 3

2 Literature Review **5**
- 2.1 Deep Learning for Image Segmentation . . . 5
  - 2.1.1 Definition and Historical Perspective . . . 5
  - 2.1.2 Fundamentals . . . 7
  - 2.1.3 U-Net . . . 9
  - 2.1.4 SegNet . . . 14
  - 2.1.5 Deeplabv3 . . . 14
- 2.2 Challenges of Underwater Vision . . . 16
- 2.3 Critical Analysis . . . 19

3 Image Segmentation for Marine Growth Prediction **21**
- 3.1 A Dataset for Marine Growth Segmentation . . . 21
- 3.2 Mitigating the Impacts of Underwater Challenges . . . 24
  - 3.2.1 Localized Cropping for Image Segmentation . . . 24
  - 3.2.2 On-the-fly Data Augmentation . . . 26
- 3.3 Learning-based architecture for Marine Growth Segmentation . . . 26

4 Experimental Results **29**
- 4.1 Experimental setup . . . 29
- 4.2 Marine Growth Segmentation . . . 32
  - 4.2.1 Initial Dataset Experiments . . . 32
  - 4.2.2 Expanded Dataset Experiments . . . 33
- 4.3 Testing in Real World Scenario . . . 35
- 4.4 Conclusion . . . 36

5 Conclusion and Future Work **39**

References **41**

List of Figures

1.1 Five step marine growth removal process . . . 2

2.1 Semantic segmentation example, bottom left is the original image, top left is the segmentation result where each type of object is identified by a different color, right is the overlap of both images. . . . 6
2.2 Convolution example . . . 8
2.3 Max (left) and Average (right) pooling examples . . . 9
2.4 U-Net architecture . . . 10
2.5 VGG16 architecture . . . 11
2.6 Residual block . . . 12
2.7 Transposed convolution . . . 13
2.8 SegNet architecture . . . 14
2.9 U-Net vs SegNet results on single class (HD, WR, RO, RI, FV) prediction and combined results . . . 15
2.10 Convolution (a) vs Atrous Convolution (b) . . . 15
2.11 Light scattering and absorption examples . . . 16
2.12 (a) Image before CLAHE, (b) CLAHE application. . . . 17
2.13 Data Augmentation examples. . . . 18

3.1 Sample images from the dataset. . . . 22
3.2 Distribution of species occurrence in the dataset. . . . 22
3.3 Contours of species present in sample (a) of figure 3.1, *Flustra foliacea* (d), *Securiflustra securifrons* (e), *Ascidiacea* (f) and the finished segmentation mask (g). . . . 23
3.4 Different types of images: blue-green toned images (a) and brown toned images (b). 23
3.5 Custom transformation generating a new image-mask pair. . . . 25
3.6 Sample image (a) generating 3 different images, (b), (c) and (d) via online data augmentation. . . . 27

4.1 Confusion matrix . . . 30
4.2 Sample segmentation mask (a) and dummy prediction (b) . . . 31
4.3 IoU visual example. . . . 31
4.4 DC Loss (top) and IoU curves (bottom) for the train (left) and test (right) sets of the initial dataset. In blue $ResNet_{512}$, in red $DeeplabV3_{512}$, in green $VGG16_{224}$ and in black $VGG16_{512}$. . . . 33
4.5 Qualitative analysis of 6 samples from the original dataset. Columns from left to right are original RGB, ground truth and $DeeplabV3_{512}$, $ResNet_{512}$, $VGG16_{224}$, $VGG16_{512}$ predictions. . . . 34

4.6 DC Loss (top) and IoU curves (bottom) for the train (left) and test (right) sets of the expanded dataset. In blue $ResNet_{512}$, in red $DeeplabV3_{512}$, in green $VGG16_{224}$ and in black $VGG16_{512}$. 35
4.7 Qualitative analysis of 6 samples from the expanded dataset. Columns from left to right are original RGB, ground truth and $DeeplabV3_{512}$, $ResNet_{512}$, $VGG16_{224}$, $VGG16_{512}$ predictions. 36
4.8 Visual analysis on new data. 37

List of Tables

3.1 Marine growth species occurrences. 24

4.1 Quantitative performance on the initial dataset, better performance is characterized by higher IoUs and lower DC Losses. 32
4.2 Quantitative results for the expanded dataset better performance is characterized by higher IoUs and lower DC Losses. 34

Abbreviations

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DL	Deep Learning
ROV	Remotely Operated Vehicle
GPU	Graphics processing units
CPU	Central processing unit
DNN	Deep Neural Network
NN	Neural Network
FCNN	Fully Convolutional Networks
CLAHE	Contrast-limited adaptive histogram equalization
ROI	Region of Interest
IEA	International Energy Agency
MG	Marine Growth
SGD	Stochastic Gradient Descent
ASPP	Atrous Spatial Pyramid Pooling
GAN	Generative Adversarial Networks

Chapter 1

Introduction

1.1 Context and Motivation

In recent years, Artificial Intelligence (AI) advancements have sparked a revolution with practical applications across all industries. Deep Learning (DL) algorithms are at the core of these advancements enabling autonomous systems to perform tasks that previously required human intervention and sometimes with increased proficiency [1]. Computer Vision (CV), a subset of AI that aims to understand and interpret visual data to extract meaningful insights, powered by these advances in AI is enabling the development autonomous systems capable of facial recognition [2], autonomous driving [3] or medical imaging analysis [1]. Leveraging these advancements the marine industry has embraced the potential of CV and is already benefitting from it's applications [2, 3, 4]. In particular, CV has proven to be a valuable tool in the field of marine maintenance [5], enabling efficient and proactive monitoring of critical assets in marine environments [6, 7, 8].

Marine environments encompass vast bodies of water, including oceans, seas, and lakes, which harbor diverse ecosystems and provide a vital resource for various industries. Within these environments, offshore structures play a pivotal role in supporting activities such as renewable energy generation, oil and gas exploration. According to the International Energy Agency (IEA), in 2021 offshore oil drilling accounted for roughly 25% of global oil production[4]. Offshore structures, such as offshore platforms, wind farms, and underwater pipelines, face unique challenges due to their exposure to harsh conditions, including strong winds, unpredictable weather patterns, corrosive saltwater and biofouling [9, 10]. In particular, biofouling or *marine growth* (MG) refer to the accumulation of marine organisms, such as, algae, barnacles and mollusks, on the surface of submerged structures such as, ship hulls, offshore structures or marine equipment. It is a natural process, although undesirable because it leads to increased drag and fuel consumption for ships, reduced efficiency of underwater structures, corrosion [11], and the introduction of invasive species

[1] https://www.bbc.com/news/technology-40042581
[2] https://www.vision-box.com/
[3] https://www.tesla.com/autopilot
[4] https://www.iea.org/reports/world-energy-outlook-2022

to new habitats. Historically, companies in the Oil & Gas industry have used divers to remove marine growth from offshore structures, and more recently, around 2 decades ago, companies started using Remote Operated Vehicles (ROVs) to substitute divers performing marine growth removal [12, 13]. However, the effectiveness of this solution is hindered by 2 main factors: high cost of acquiring and maintaining these vehicles and poor control of the ROV [14, 15]. Pedersen *et al.* (2022) [15] present a five step process to marine growth removal (Figure 1.1):

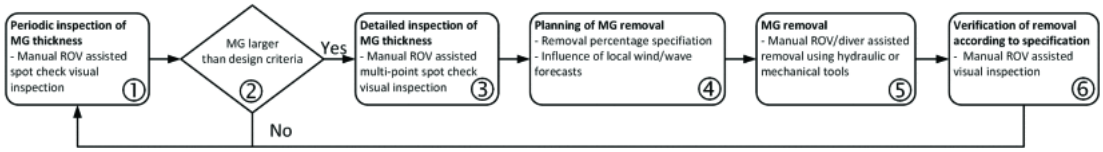

Figure 1.1: Five step marine growth removal process (extracted from [15]).

In step 1, regular inspections of the structure are conducted using ROVs equipped with cameras and sensors to gather data on the marine growth present. This data is then analyzed by experts in step 2, who quantify the extent of the marine growth and make a decision regarding maintenance actions. If maintenance is required, the process moves to step 3, otherwise, the process is postponed until the next scheduled assessment. In step 3, a more detailed inspection is carried out in the affected area and in step 4 plans are made for the upcoming removal campaign. Step 5 involves the actual removal of marine growth, which is performed by an ROV equipped with a high-pressure water jet [16]. Finally, in step 6, a final inspection is conducted to assess the effectiveness of the cleaning process. This sequential approach ensures systematic monitoring, assessment, and targeted removal of marine growth from offshore structures.

The objective of this work is to automate the decision process in step 2. The current decision-making process is manual and time-consuming, to address this, this work proposes the utilization of state-of-the-art CV algorithms to develop an autonomous system capable of accurately identifying and delineating regions of marine growth within underwater images.

In recent years, significant advancements in CV have demonstrated promising results, with performance approaching or even surpassing human-level capabilities in certain tasks [17]. One crucial task within the CV domain is image segmentation, which involves the partitioning of an image into distinct regions based on shared visual characteristics, such as color, texture, or intensity. Specifically, image segmentation entails assigning a label to each pixel in an image, ensuring that pixels with the same label exhibit similar characteristics. The primary objective of image segmentation is to accurately separate different regions and identify object boundaries.

1.2 Objectives

The proposed research aims to contribute to the advancement of underwater autonomous systems by leveraging these CV techniques. Overall, the primary goals of this work are as follows:

- Improve inspection capabilities for offshore maintenance by developing a DL-based architecture to perform image segmentation on underwater imagery in order to identify regions of marine growth.
- Perform a rigorous comparative analysis of the models utilizing appropriate quantitative metrics, with the aim of objectively assessing their performance and discerning any potential variations or advancements in their segmentation capabilities.
- Benchmark the developed models with new data acquired in real scenarios in order to test the generalization capabilities.

1.3 Work structure

This work aims to provide a comprehensive analysis on the use of Deep Learning image segmentation algorithms in an underwater context. To achieve this, the following sections will be covered:

- Chapter 2 contains a section about deep learning for image segmentation, referring most popular architectures with their advantages and disandvantages, and a section containing information about the challenges on underwater vision and a review on the literature proposed to overcome them. The chapter ends with a critical analysis of the literature and a discussion on how it relates to this specific work.
- Chapter 3 explains the specific methods employed to address the problem questions and achieve the intended goals. It contains a detailed description of the data collection, preprocessing and augmentation, the networks used along with the chosen hyperparameters for training. The section also highlights any limitations or potential challenges encountered during the work.
- Chapter 4 presents the findings obtained from the analysis conducted in the previous section. It includes comparison of the trained models by different evaluation metrics. It also includes visual comparisons of the results. The section focuses on providing a comprehensive and objective interpretation of the findings in relation to the research objectives.
- Chapter 5 presents a summary of the key findings of the study alongside a discussion about the significance and implications of the findings. Additionally, any limitations of the study are addressed and acknowledged.

Chapter 2

Literature Review

This chapter centers on the utilization of cutting-edge CV algorithms within the underwater context, specifically focusing on image segmentation. The chapter commences with a theoretical exploration of deep learning techniques in the realm of image segmentation. It encompasses an introduction to the image segmentation task, encompassing both classical and modern methods employed. Subsequently, prominent deep learning architectures are introduced, elucidating their underlying principles, advantages, and limitations. The following section delves into the examination of related work that incorporates deep learning algorithms for image segmentation in the underwater domain. It explores the techniques employed by various researchers to enhance the performance of these algorithms within an underwater setting. The aim is to provide a comprehensive overview of the existing literature and shed light on the advancements made in utilizing deep learning for underwater image segmentation.

2.1 Deep Learning for Image Segmentation

2.1.1 Definition and Historical Perspective

Image segmentation is a fundamental task in computer vision that involves partitioning an image into distinct regions or segments based on specific criteria. The main objective is to assign a label or category to each pixel or group of pixels in the image, enabling the differentiation of various objects or regions of interest. This process facilitates the extraction of meaningful information, allowing for accurate analysis, understanding, and interpretation of visual data. Image segmentation encompasses three primary types: semantic segmentation, instance segmentation, and panoptic segmentation. **Semantic segmentation**, the focus of this work, aims to detect the class or category to which each pixel belongs (Figure 2.1). **Instance segmentation** goes a step further by identifying the specific instance or occurrence of an object for each pixel, essentially detecting and differentiating individual objects within the image. **Panoptic segmentation** combines the principles of both semantic and instance segmentation, providing class identification for each pixel while also distinguishing separate instances of the same class.

Figure 2.1: Semantic segmentation example, bottom left is the original image, top left is the segmentation result where each type of object is identified by a different color, right is the overlap of both images.

Classical methods to perform image segmentation include:

- **Thresholding**: a technique that converts a grayscale image into a binary image by applying a clip-level or threshold value [18][19]. The primary objective of thresholding is to accurately select the optimal threshold value or values when multiple levels are involved. In the context of industry applications, a commonly employed method is Otsu's method [20]. This method determines the threshold by minimizing the intra-class intensity variance. By analyzing the distribution of pixel intensities in the grayscale image, Otsu's method identifies the threshold that maximizes the separation between object and background, resulting in an effective binary image representation;

- **Clustering**: these algorithms aim to identify distinct clusters or groups within an image based on similarities in color, texture, intensity, or other feature descriptors. One popular clustering algorithm used for image segmentation is the K-means algorithm [21]. K-means partitions the image pixels into K clusters, where K is a predefined number. It iteratively assigns pixels to clusters based on the proximity to cluster centroids and updates the centroids until convergence.

- **Edge detection**: algorithms that aim to locate areas of significant intensity transitions, which often correspond to object boundaries, edges, or discontinuities in the image [22]. One commonly used method is the Canny edge detection algorithm [23], which involves multiple stages, including noise reduction, gradient calculation, non-maximum suppression, and hysteresis thresholding. The Canny algorithm produces high-quality edges by suppressing noise and detecting true edges with subpixel accuracy.

- **Region growing**: these are algorithms based on the concept of region connectivity. They aim to group pixels or regions together that have similar values [24][25]. The region growing process starts with the selection of one or more seed points or seed regions, which serve

as the initial regions of interest. These seeds are iteratively expanded by incorporating neighboring pixels or regions that satisfy certain similarity criteria [26]. The criteria can vary depending on the specific algorithm and application but commonly include intensity similarity, color similarity, or spatial proximity. As the algorithm progresses, neighboring pixels or regions are recursively added to the growing region until the similarity criteria are no longer met.

Modern methods of image segmentation have witnessed significant advancements due to the rise of DL and convolutional neural networks (CNNs). Unlike traditional algorithms, which often require manual feature extraction and preprocessing steps [27], DL can learn useful features and representations directly from the data, offering end-to-end solutions. CNNs are designed to automatically learn hierarchical features at different levels, starting from low-level edges and textures to high-level object representations [28][29]. This eliminates the need for explicit feature engineering, as the model learns to extract the most relevant features for the given task.

2.1.2 Fundamentals

CNNs are a class of deep learning models specifically designed for processing grid-like data such as images. They are highly effective in capturing and extracting hierarchical patterns and features from input data. Krizhevsky *et al* (2012) [30] proposed the use of CNNs, networks mainly composed of convolutional layers for image classification in what became commonly known as the **AlexNet** paper. Their network achieved groundbreaking results on the **ImageNet dataset** [31], which is a large-scale dataset for image classification that is used as a benchmark for image classification algorithms. AlexNet contained convolutional layers, pooling layers and fully connected layers and the success of the architecture paved the way for subsequent advancement in the field. The main components of a CNN include convolutional layers, pooling layers and fully connected (FC) layers.

2.1.2.1 Convolutional layer

Convolutional layers are designed to extract local spatial patterns and capture hierarchical representations from input data such as images. The key operation in convolutional layers is convolution, where small filters or kernels slide over the input data, computing element-wise multiplications and summations. This process enables the layer to capture local correlations and detect features regardless of their spatial position. By stacking multiple convolutional layers, CNNs can learn increasingly abstract and complex representations, making them highly effective in tasks such as image classification, object detection, and image generation. The inherent architecture and operations of convolutional layers allow them to leverage the spatial relationships in the data, making them a fundamental building block for successful computer vision applications.

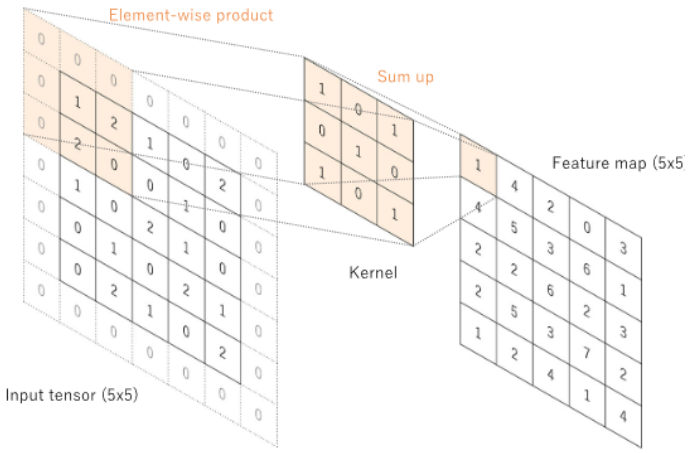

Figure 2.2: Convolution example (extracted from [32].

2.1.2.2 Pooling layer

Pooling layers are used to summarize feature maps generated by the convolutional layers. Their main purpose is to reduce spatial dimensionality of the input while preserving the most relevant features, they do this by dividing the input into smaller pieces and using a mathematical function to obtain a single value that is representative of the region [30]. The most popular functions that do this are average pooling and max pooling. In average pooling, the value that represents the region is the average of all the values contained within that region, this is useful because it provides a general representation of the whole region since every value is represented. In max pooling the value that represents the region is the maximum value of all the values contained within that region, this means that only the most dominant feature is preserved after this operation. The key benefits of pooling layers are:

- **Enhanced robustness:** by only capturing the most relevant features, pooling makes the network more resistant to shifts or distortions.
- **Reducing overfitting:** learning only the most salient features allow the network to abstract and generalize more from the data it is given.
- **Capturing invariance:** by dividing the input into smaller regions, the network learns to recognize objects independent of their location within the input.

2.1.2.3 Fully Connected layer

In a FC layer, each neuron is connected to every neuron in the previous layer and to every neuron in the next layer. This allows for unrestricted information flow and enables the layer to learn complex relationships between features. In CNNs, FC layers are typically placed after one or

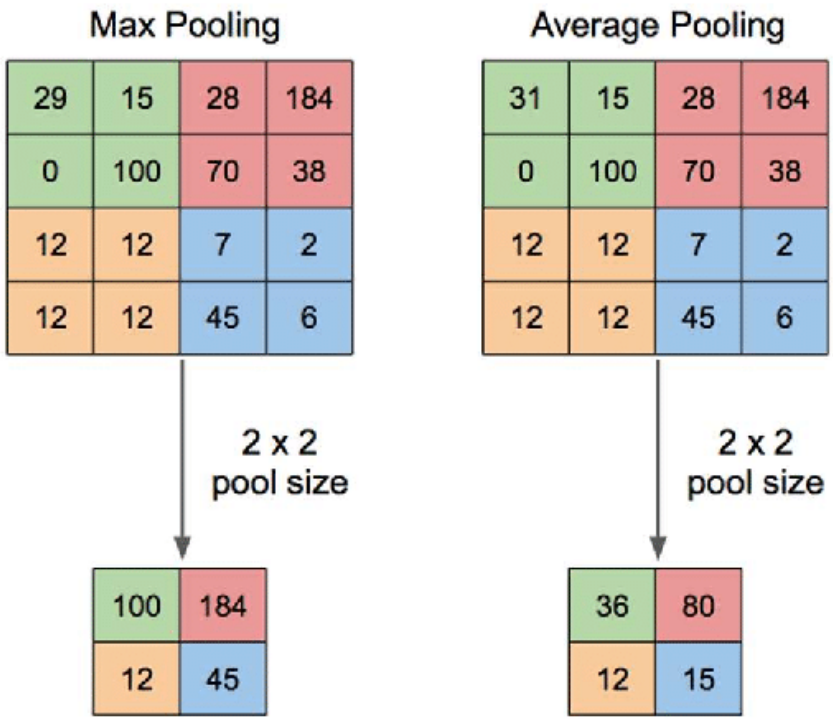

Figure 2.3: Max (left) and Average (right) pooling examples (extracted from [33]).

more convolutional and pooling layers. The output of the preceding layers, often in the form of a flattened or reshaped feature map, is fed into the FC layer. Each neuron in an FC layer is associated with a weight parameter that determines the strength of its connection to the previous layer. During the training phase, these weights are adjusted based on the backpropagation [34] algorithm, which computes the gradients of the loss function with respect to the network parameters and updates them accordingly. This optimization process aims to minimize the error or loss of the network's predictions. FC layers play a crucial role in learning complex combinations of features from the extracted representations in the earlier layers. They enable the network to capture high-level patterns and relationships among the learned features, ultimately leading to better discrimination and classification performance. The activation function used in FC layers introduce non-linearity into the network, allowing it to learn and model non-linear relationships in the data. Due to the unrestricted connectivity they usually have high amounts of *parameters* and are prone to overfitting [35].

2.1.3 U-Net

The U-Net is a popular CNN architecture commonly used for image segmentation tasks. It was introduced by Ronneberger *et al.* (2015) [36] as a specifically designed network for biomedical image segmentation by leveraging an encoder-decoder structure with skip connections (Figure 2.4). The encoder, also referred as the backbone, features a series of convolutional and pooling

layers. These layers gradually reduce the spatial dimensions of the input image while increasing the number of feature channels. This process allows the network to learn high-level representations of the input image. The backbone of the network may have implementations of popular CNN architectures like ResNet [37] or VGG16 [38] due to their strong performance in image classification tasks and their ability to capture high-level features. The decoder part of the U-Net is an upsampling path that aims to recover the spatial resolution lost during the encoding process. Each upsampling step consists of an upsampling operation followed by a concatenation with feature maps from the corresponding encoding path. This concatenation is a skip connection that allows the network to utilize both low-level and high-level features, aiding in precise localization. By incorporating skip connections, the network can merge both local and global information, enabling accurate localization of objects.

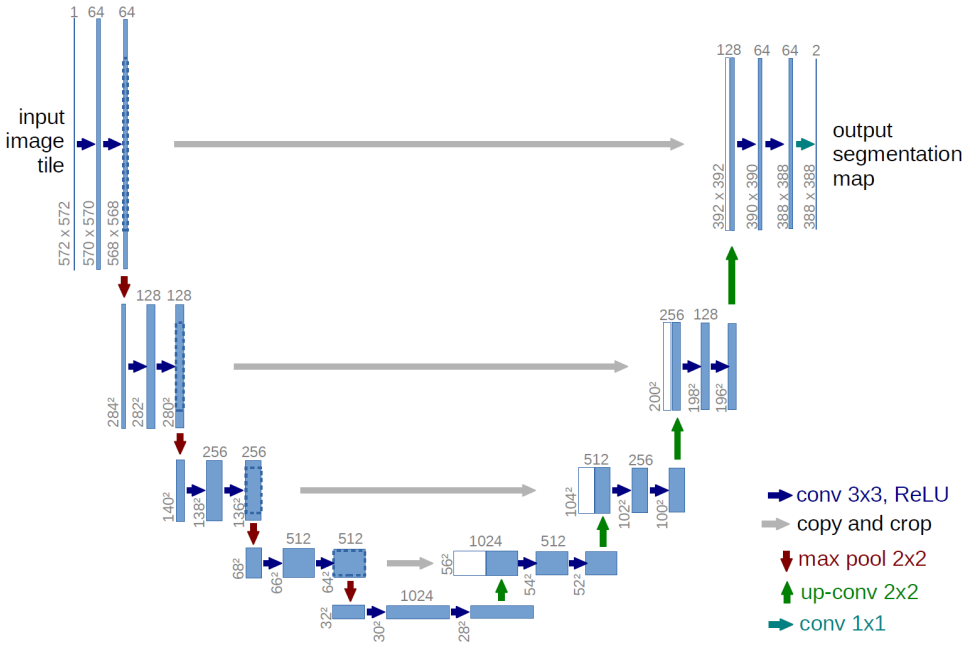

Figure 2.4: U-Net architecture [36].

2.1.3.1 Encoder: VGG

The VGG network, short for Visual Geometry Group network, is a widely recognized CNN architecture introduced by Simonyan *et al* (2014) [38]. VGG is known for its simplicity and effectiveness, offering a straightforward and easy-to-understand architecture for image classification tasks. The key characteristic of the VGG network is its uniform structure, where the convolutional layers consist of small 3x3 filters throughout the entire network. This design choice allows for deeper networks to be trained while keeping the network architecture simple and manageable. VGG architectures typically vary in depth, with the original VGG network offering 16 convolutional layers (VGG16, Figure 2.5) or 19 convolutional layers (VGG19). VGG networks utilize a

series of convolutional layers with ReLU activations, followed by max-pooling layers to reduce spatial dimensions. The final layers of the VGG network usually consist of fully connected layers, leading to a softmax layer for classification. VGG networks are trained using stochastic gradient descent (SGD) with weight decay and dropout regularization techniques to prevent overfitting.

Despite its simplicity, the VGG network has achieved remarkable performance in various image recognition tasks, particularly in large-scale image classification challenges such as the ImageNet dataset [31]. The uniform structure of VGG enables it to learn hierarchical representations of images, capturing both low-level and high-level features effectively. The deep layers of the VGG network allow it to learn more complex representations, resulting in improved discriminative capabilities.

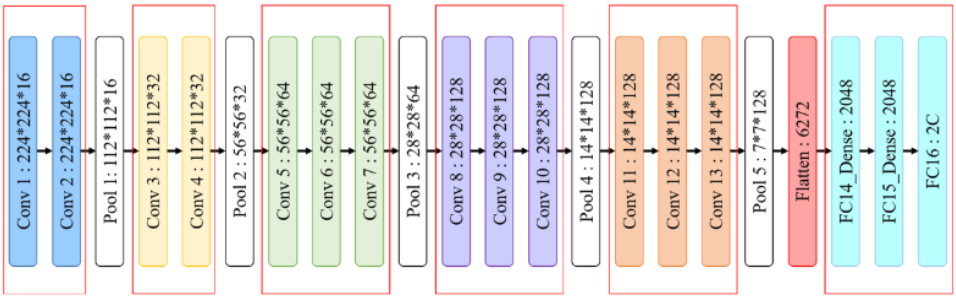

Figure 2.5: VGG16 architecture (extracted from [39]).

The VGG network has several variations, primarily based on the number of layers in the network. Here are the different types of VGG networks:

1. VGG16: The VGG16 network consists of 16 convolutional layers. It starts with a series of convolutional layers with 3x3 filters, followed by max-pooling layers for downsampling. The architecture then includes three fully connected layers leading to the final softmax layer. VGG16 gained popularity as one of the early deep CNN architectures and has been widely used in various image classification tasks.

2. VGG19: The VGG19 network extends VGG16 by adding three additional convolutional layers, resulting in a total of 19 convolutional layers. The extra layers contribute to increased model complexity and potentially improved performance.

3. Other Variations: In addition to VGG16 and VGG19, researchers have explored variations of the VGG network with different depths and configurations. For example, VGG11 and VGG13 are shallower versions of VGG16, containing 11 and 13 convolutional layers, respectively. These lighter versions are useful when computational resources are limited, as they offer a trade-off between model complexity and performance.

It's worth noting that the primary distinction between these variations lies in the number of layers, while the overall architecture and design principles remain the same. The uniformity in

architecture across VGG networks has made them easily understandable and adaptable for experimentation and research purposes. The VGG network has also played a significant role in the development of deep learning research. It has served as a baseline model for benchmarking and comparison against more complex architectures. Researchers have used VGG as a starting point for fine-tuning or transfer learning in various domains, allowing for the application of pre-trained VGG models on different image recognition tasks. While VGG networks are computationally more expensive due to their depth and the use of 3x3 filters throughout, their simplicity and strong performance make them a valuable tool in the deep learning toolbox. Researchers and practitioners continue to explore and build upon the ideas introduced by the VGG network, influencing subsequent developments in CNN architectures and their applications in computer vision.

2.1.3.2 Encoder: ResNet

ResNet, short for Residual Network, is a groundbreaking CNN architecture introduced by He *et al.* (2015) [37] that addresses the challenge of training very deep neural networks by mitigating the vanishing gradients problem, where accuracy saturates or even degrades as networks become deeper [40]. The key innovation in ResNet is the introduction of residual blocks, which allow for the learning of residual or residual-like mappings. Unlike traditional CNNs, residual blocks employ skip or shortcut connections that bypass one or more layers. These connections enable the network to learn residual functions, capturing the difference between the desired output and the current output of the network. By propagating the error through these shortcut connections, ResNet effectively enables the training of extremely deep networks without degrading accuracy.

The residual blocks in ResNet are typically composed of convolutional layers, with batch normalization and ReLU activation functions (Figure 2.6). The architecture also includes global average pooling and a fully connected layer at the end for classification. Various versions of ResNet have been proposed, such as ResNet-18, ResNet-34, ResNet-50 (Figure **??**), ResNet-101, and ResNet-152, which differ in the number of layers and the complexity of the network.

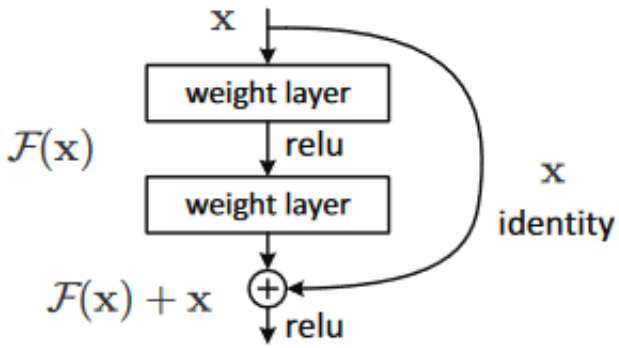

Figure 2.6: Residual block (extracted from [37]).

ResNet has had a profound impact on the field of deep learning. Its introduction of skip connections revolutionized the way deep neural networks are designed and trained. ResNet's architecture has achieved state-of-the-art results in various computer vision tasks. By enabling the training of deep networks, ResNet has paved the way for deeper architectures that can learn more complex representations and achieve higher performance.

2.1.3.3 Decoder

The decoder module plays a crucial role in the process of upsampling and reconstructing the feature maps to obtain a high-resolution output. After the initial downsampling steps in the encoder module, the decoder module starts with the lowest resolution feature maps and gradually upsamples them using transposed convolutions. These operations perform element-wise between the convoluted feature maps produced by the encoder module and kernels with values optimized during training (Figure 2.7). We can calculate an output of size $O\,x\,O$ of a transposed convolution using the following formula, given an input feature map of size $I\,x\,I$, kernel size $K\,x\,K$, stride s, and padding p:

$$O = (I-1)*s-2*p-(K-1)-1$$

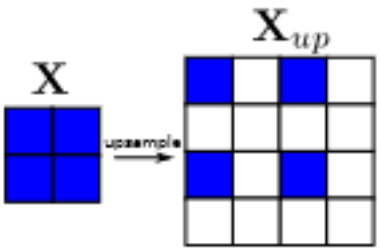

Figure 2.7: Transposed convolution (extracted from [41]).

The decoder module typically consists of a series of upsampling blocks, where each block combines the upsampled feature maps with the corresponding feature maps from the encoder module. This skip-connection allows the decoder to leverage both low-level and high-level information, aiding in the precise localization of objects and maintaining fine-grained details. Each upsampling block in the decoder includes additional convolutional layers for feature refinement and dimensionality reduction [42]. The decoder progressively expands the spatial dimensions while refining the feature representations, leading to a reconstructed output that closely resembles the input image in terms of resolution and semantic content.

2.1.4 SegNet

SegNet is a deep learning architecture specifically designed for image segmentation tasks, introduced by Badrinarayanan *et al* (2015) [43]. SegNet features an econder-decoder structure similar to the U-Net. The encoder module consists of multiple convolutional and pooling layers, gradually reducing the spatial dimensions of the input image while extracting hierarchical features. The decoder module, on the other hand, performs upsampling of the low-resolution feature maps to recover the original input size. What makes SegNet unique is its utilization of pooling indices obtained during the encoding phase in its skip connections, as displayed in Figure 2.8, which are then used for precise pixel-wise upsampling in the decoder module. This approach enables SegNet to retain important spatial information and produce accurate segmentation results.

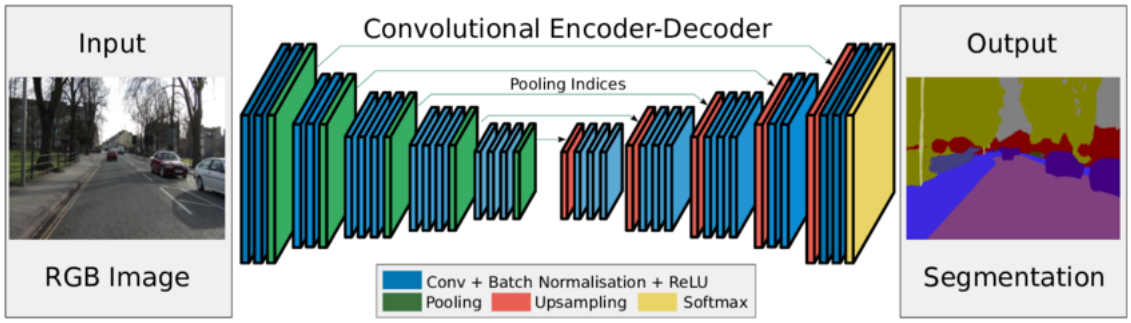

Figure 2.8: SegNet architecture (extracted from [43]).

The main difference between U-Net and SegNet lies in their computacional consumption. SegNet reuses memorized pooling indices from the encoder during the upsampling, while U-Net transfers the entire feature maps for upsampling. The practice of re-utilizing previously memorized values effectively saves memory in the system [44]. In terms of performance the work of Islam *et al* (2020) [45] shows U-Net outperforming SegNet for underwater segmentation. The authors present a novel general-purpose dataset for underwater segmentation and evaluate the performance of multiple popular architectures, among them the U-Net and SegNet. Figure 2.9 shows SegNet achieving comparable performance when given a powerful feature extractor like the ResNet and U-Net is fed grayscale images and U-Net outperforming SegNet when given RGB images as input.

2.1.5 Deeplabv3

DeepLabv3 is a highly influential CNN architecture developed for semantic image segmentation proposed by Chen *et al* (2018) [46]. DeepLabv3 builds upon the success of its predecessors, DeepLab [47] and DeepLabv2 [48], and introduces several key innovations that improve its accuracy and efficiency. The core contribution of DeepLabv3 lies in its use of atrous (dilated) convolutions, which allow for multi-scale feature integration without significantly increasing the

	Model	HD	WR	RO	RI	FV	Combined
$\mathcal{F}$	SegNet_{CNN}	59.60 ± 2.02	41.60 ± 1.65	31.77 ± 3.03	41.88 ± 2.66	60.08 ± 1.91	46.97 ± 2.25
	SegNet_{ResNet}	80.52 ± 3.26	77.65 ± 3.15	62.45 ± 3.90	82.30 ± 1.96	91.47 ± 1.01	76.88 ± 2.66
	UNet_{GRAY}	85.47 ± 2.21	79.77 ± 2.01	60.95 ± 3.31	69.95 ± 2.57	84.47 ± 1.39	75.12 ± 2.30
	UNet_{RGB}	89.60 ± 1.84	86.17 ± 1.73	68.87 ± 3.30	79.24 ± 2.70	91.35 ± 1.14	83.05 ± 2.14
$mIOU$	SegNet_{CNN}	62.76 ± 2.35	66.75 ± 2.57	36.63 ± 3.12	63.46 ± 3.18	62.48 ± 2.32	58.42 ± 2.71
	SegNet_{ResNet}	74.00 ± 2.88	82.68 ± 2.94	58.63 ± 3.61	89.61 ± 1.15	82.96 ± 1.38	77.58 ± 2.39
	UNet_{GRAY}	78.33 ± 2.34	85.14 ± 2.14	57.25 ± 3.00	79.96 ± 2.55	78.00 ± 1.90	75.74 ± 2.38
	UNet_{RGB}	81.17 ± 2.02	87.54 ± 2.00	62.07 ± 3.12	83.69 ± 2.58	83.83 ± 1.47	79.66 ± 2.24

Figure 2.9: U-Net vs SegNet results on single class (HD, WR, RO, RI, FV) prediction and combined results (adapted from [45]).

computational cost. By applying atrous convolutions at multiple rates, DeepLabv3 captures both fine-grained details and global context, enabling precise and comprehensive segmentation. The network employs a deep backbone network, such as ResNet, to extract high-level feature representations. These features are then refined using atrous spatial pyramid pooling (ASPP), which involves parallel atrous convolutions at different rates to capture multi-scale contextual information (Figure 2.10).

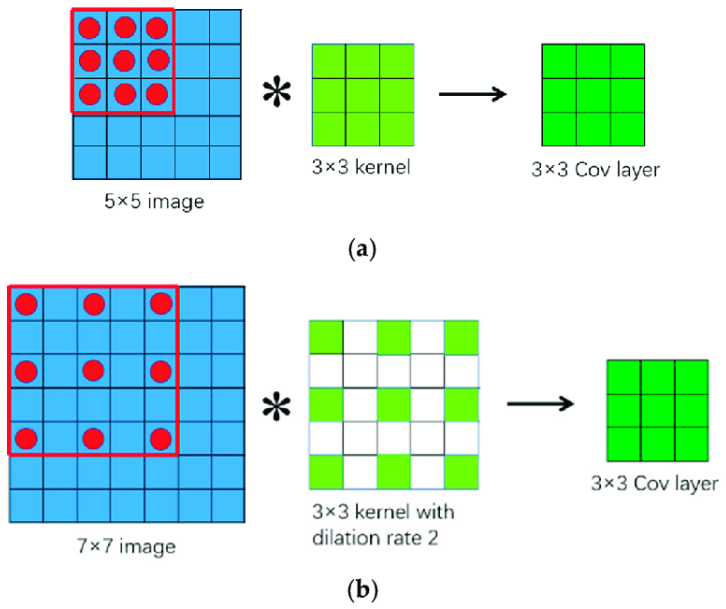

Figure 2.10: Convolution (a) vs Atrous Convolution (b) (exctrated from [49]).

DeepLabv3 also incorporates a skip connection module that combines high-resolution features from earlier stages of the network with the ASPP module's multi-scale features. The skip connections help in preserving and integrating fine-grained spatial information, facilitating more accurate localization of object boundaries. Another notable aspect of DeepLabv3 is its use of dilated convolution in the final prediction layer. This allows the network to generate pixel-level predictions

at the original image resolution, avoiding the need for upsampling. By maintaining the resolution, DeepLabv3 produces more precise segmentation results.

DeepLabv3 has demonstrated state-of-the-art performance in various challenging semantic segmentation benchmarks, including PASCAL VOC [50] and Cityscapes [51] datasets. Its ability to capture fine details, exploit multi-scale context, and leverage skip connections has made it highly effective in segmenting objects of varying scales and complex structures.

2.2 Challenges of Underwater Vision

The development of underwater vision systems encounters two significant challenges that pose technical complexities. Firstly, the acquisition of underwater data is hindered [52] by the requirement for specialized vehicles and equipment. Underwater exploration typically relies on ROVs or Autonomous Underwater Vehicles (AUVs) [53, 54], which come with their own logistical and operational considerations. These vehicles need to be equipped with underwater cameras and sensors capable of capturing high-quality data in a challenging aquatic environment [55, 56]. The design, deployment, and maintenance of these systems involve significant technical expertise and infrastructure. Secondly, underwater conditions introduce various factors that degrade image quality, thereby impeding the effectiveness of vision systems [57]. Light refraction and scattering phenomena in water result in reduced visibility and distortion of images. As light travels through water, it interacts with suspended particles, dissolved substances, and organisms, leading to absorption and scattering effects as demonstrated in Figure 2.11. This causes the captured images to suffer from decreased contrast, color shifts, and blurring.

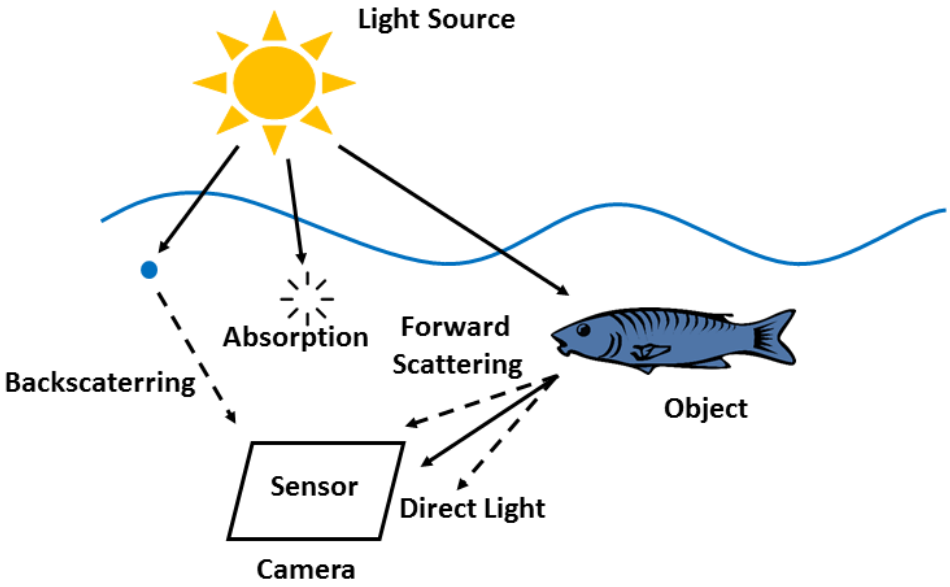

Figure 2.11: Light scattering and absorption examples (extracted from [58]).

To migitate these effects researchers have developed image enhancement methods that increase the quality of the images before sending them for further processing. In the work of Want *et al.* (2022) [59], the authors present a novel algorithm that incorporates a series of color correction operations to address the adverse effects of light refraction and other underwater conditions on image quality. The proposed algorithm employs various techniques, including white balancing, γ correction, contrast-limited adaptive histogram equalization (CLAHE), bilateral filtering, and single-scale retinex. These operations are specifically designed to alleviate color distortions, enhance contrast, and improve overall image clarity in underwater environments. By sequentially applying these color correction techniques, the algorithm effectively mitigates the detrimental effects of light refraction and other underwater conditions, resulting in visually improved and more accurately representational images for further analysis and processing.

Histogram equalization is a widely employed technique in image processing for enhancing the contrast and improving the overall appearance of digital images. It aims to redistribute the pixel intensities across the entire dynamic range, effectively stretching the histogram to span the full extent of available intensity levels. By equalizing the histogram, the resultant image exhibits a more balanced distribution of intensities, leading to enhanced details and increased visual distinguishability of objects and structures. CLAHE is an advanced variation of histogram equalization developed to address the limitations of the traditional method, mainly, over-brightness and loss of information because the histogram is not limited to a particular region. CLAHE addresses this by performing histogram equalization on small blocks of the image called *tiles* and limiting the amount of contrast allowed in each region to prevent over-brightness. In this work, however, the CLAHE algorithm was not applied due to leading to an improper highlight of image objects. As evidenced in Figure 2.12, the CLAHE algorithm not only increases contrast and highlight on the marine growth object located in the bottom right corner, but also, the rest of the image. This phenomenon happens in images with large homogeneous zones, similar to the brown-toned images that comprise most of the dataset, described in the previous section.

Figure 2.12: (a) Image before CLAHE, (b) CLAHE application.

Zhou *et al* (2019) [60] propose a GAN-based image enhancement technique. Generative Adversarial Networks (GANs) are a type of neural network that has revolutionized generative modelling. They excel at image generation and are capable of generating high quality images that

resemble real ones. The authors leverage this ability to generate good quality images from blurry images which have their quality affected by the underwater environment, effectively diminishing the impact of the underwater environment on their dataset.

Alongside image enhancement techniques researchers use data augmentation techniques to artificially increase their datasets. Data augmentation is a technique widely used in deep learning to increase the size and diversity of training datasets by applying various transformations to existing data. It aims to enhance the generalization and robustness of models by exposing them to a broader range of variations and patterns in the data. Data augmentation is particularly valuable when the available training data is limited or imbalanced, as it effectively expands the dataset without requiring additional data collection. The application of data augmentation involves systematically modifying the input data while preserving the label or ground truth. Common augmentation techniques include random rotations, translations, scaling, flipping, cropping, and adding noise or distortions to the images (Figure 2.13).

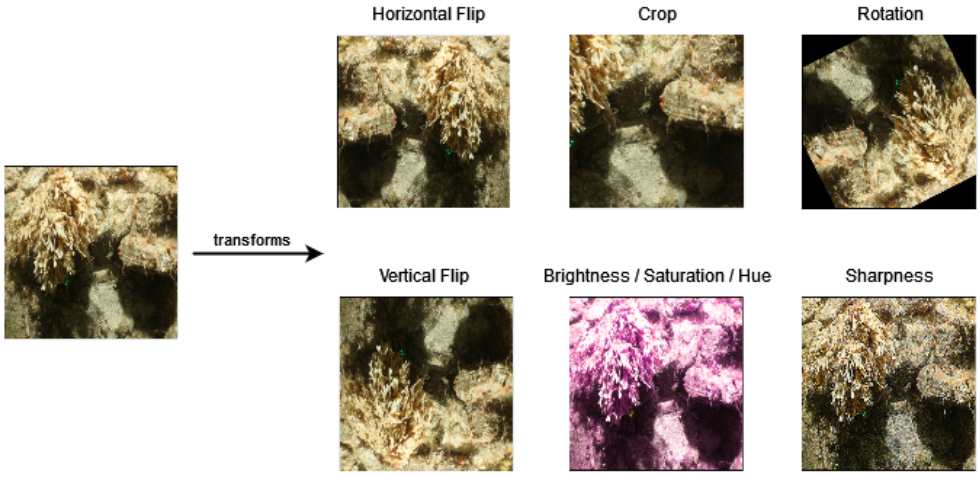

Figure 2.13: Data Augmentation examples.

In the work of Drews-Jr *et al* [61] (2021) the authors propose to increase their available data by mixing their underwater dataset with non-underwater images that have been degraded with methods based on [62] [63] to display some characteristcs of underwater images like increased turbidity, this aids the model in abstracting from these charateristics and learning features more closely related to the classes the authors are actually trying to predict. Furthermore, the authors employ transfer learning to increase their model performance. Transfer learning is a powerful technique in DL that leverages knowledge learned from one task to improve performance on a different but related task. It involves using pre-trained models, typically trained on large-scale datasets, as a starting point for a new task, instead of training a model from scratch. By transferring the learned knowledge, the model can benefit from general features and representations that are applicable to both the pre-training task and the target task, even when the datasets are different. Transfer learning offers several advantages. First, it enables the use of pre-trained models that have learned rich representations from vast amounts of data, saving significant computational resources

and time compared to training from scratch. Second, it allows models to generalize better with limited labeled data in the target task, as the pre-trained model has already learned useful features. This is useful because image segmentation tasks don't usually have high quality and acessible datasets due to the process of creating the dataset being manual and difficult. Lastly, transfer learning can help overcome the problem of overfitting, as the pre-trained model has already learned generalizable features that can benefit the target task.

2.3 Critical Analysis

In regards to the architecture used, U-Net and Deeplabv3 are the networks featured in most approaches since they have a historical track record of achieving high performance metrics and are usually at least referenced as a baseline against fine-tuned custom methods. They are versatile networks because you can switch their backbone between popular CNNs like ResNet or VGG and compare their results while maintaining the overall structure.

In regards to underwater vision challenges, it is necessary to enable models to abstract from underwater conditions that negatively impact the quality of the data. This can be achieved through image enhancement, data augmentation and transfer learning techniques. Also, to the best of the author's knowledge, no work has been found that performs segmentation specifically on marine growth, making it difficult to benchmark this work's performance with other external results. The challenges extend beyond the choice of architecture and encompass various aspects related to the quality of data and data handling processes. While the selection of an appropriate segmentation architecture is important, it is crucial to acknowledge that the performance and effectiveness of image segmentation in underwater environments heavily relies on the quality of the available data.

Chapter 3

Image Segmentation for Marine Growth Prediction

Developing a model capable of predicting marine growth in underwater images can help automate the maintenance process employed to remove marine growth from offshore structures. This chapter will cover the steps taken to generate a quality dataset capable of enabling the models being developed to reach high performance metrics. Furthermore this chapter will cover all the models trained and their specificities, along with the hyperparameters chosen for the training.

3.1 A Dataset for Marine Growth Segmentation

Image segmentation datasets require both sample images and segmentation masks to be fully functional, they will be referred as *inputs* and *labels* respectivelly. The inputs are the original images where the predictions are being made, the labels are used to annotate each pixel with a class value indicating the object or region of interest to which it belongs. The label is typically represented as a pixel-wise annotation, where each pixel is assigned a class value based on the corresponding object or region of interest. Binary masks are used for binary segmentation tasks, these are masks where every pixel is set to 1 if it belongs to the region of interest (ROI) and 0 otherwise. In multi-class segmentation tasks the label encodes more than one class, allowing the model to differentiate between different classes of objects and regions of interest. The labels are crucial for image segmentation because they represent the ground truth annotations that the model is trained to predict and their quality directly impacts the quality of the predictions.

The first task to build the dataset was to generate the segmentation masks, to do this it was verified that each image was accompanied by several image files, each containing a contour of a specific species of marine growth present in the original image, according to a marine growth specialist. A study was conducted to evaluate the amount of species present and the amount of occurrences per species present in the dataset (Table 3.1) and 25 different species were found with an average of 13.56 occurrences per species and the most frequent species in the dataset is the *Flustra foliacea* with 58 occurrences (Figure 3.2). Due to the reduced size of the dataset,

Figure 3.1: Sample images from the dataset.

performing image segmentation on 25 different classes, one for each species, would be extremely difficult, in order to circumvent this, a more general class "*marine growth*" was created that covers all the previously mentioned species. This way, the segmentation masks generated are binary meaning that a pixel with a value of **1** is within a marine growth region and a pixel with a value of **0** is located in the background.

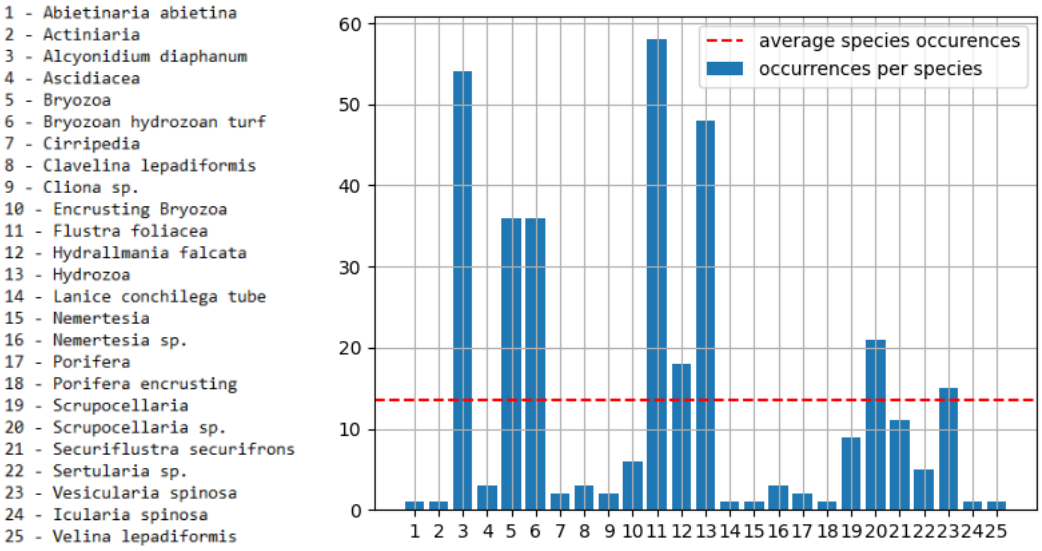

Figure 3.2: Distribution of species occurrence in the dataset, with the x-axis corresponding to the number of each species in table 3.1.

In order to generate the segmentation masks, the contours files were analyzed and processed with the aim of defining regions of marine growth using the contours as the border between regions of marine growth and background. Lastly the images for each species are overlapped resulting in a finished segmentation mask, displayed in figure 3.3. The generated dataset contains **150 underwater images** displaying different species of marine growth as displayed in Figure 3.1 with a resolution of **5184x3456** pixels. Following the mask generation process, the class distribution of the dataset was studied, having been found that:

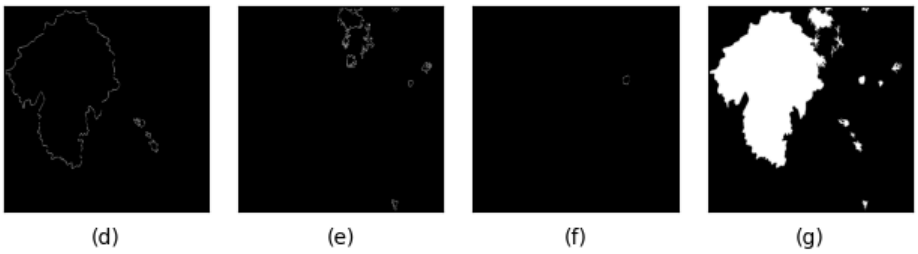

Figure 3.3: Contours of species present in sample (a) of figure 3.1, *Flustra foliacea* (d), *Securiflustra securifrons* (e), *Ascidiacea* (f) and the finished segmentation mask (g).

- **Class 0** (background) has 2.54×10^9 pixels, accounting for 94.77% of the total number of pixels in the dataset
- **Class 1** (marine growth) has 1.41×10^9 pixels, accounting for 5.23% of the total number of pixels in the dataset
- The image with the highest marine growth coverage has 25.9% of its total pixels covered by marine growth

A visual analysis of the images was conducted and 2 distinct types of images were found, as displayed in Figure 3.4. The first one consists of images with a blue-green color tone in an environment with rocks and it is around **18.7%** of the total size of the dataset; the second type consists of images with a brown color tone and a sandy environment and comprises **81.3%** of the total size of the dataset. Furthermore, the brown toned images exhibit **low variability** between themselves making them difficult to distinguish with a naked eye which can be an indication of them being prone to overfitting. In consequence, when doing the split of data for training and testing, the same distribution of blue-green images and brown images was kept in both the training and test sets.

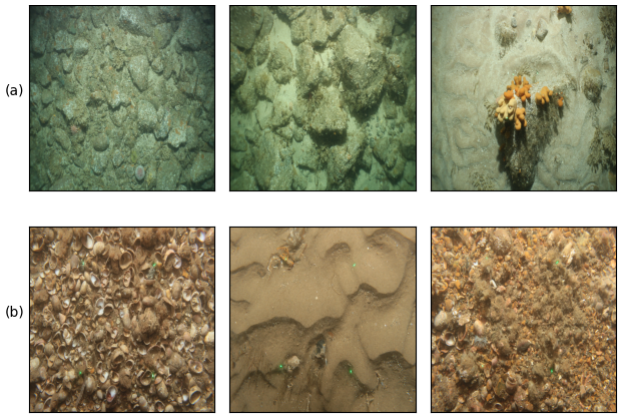

Figure 3.4: Different types of images: blue-green toned images (a) and brown toned images (b).

Table 3.1: Marine growth species occurrences.

Species	Occurrences	% of Occurrences
Abietinaria abietina	1	0.67%
Actiniaria	1	0.67%
Alcyonidium diaphanum	54	36.0%
Ascidiacea	3	2.0%
Bryozoa	36	24.0%
Bryozoan_hydrozoan turf	6	4.0%
Cirripedia	2	1.33%
Clavelina lepadiformis	3	2.0%
Cliona sp.	2	1.33%
Encrusting Bryozoa	6	4.0%
Flustra foliacea	58	38.67%
Hydrallmania falcata	18	12.0%
Hydrozoa	48	32.0%
Lanice conchilega tube	1	0.67%
Nemertesia	1	0.67%
Nemertesia sp.	3	2.0%
Porifera	2	1.33%
Porifera encrusting	1	0.67%
Scrupocellaria	9	6.0%
Scrupocellaria sp.	21	14.0%
Securiflustra securifrons	11	7.33%
Sertularia sp.	5	3.33%
Vesicularia spinosa	15	10.0%
Icularia spinosa	1	0.67%
Velina lepadiformis	1	0.67%

3.2 Mitigating the Impacts of Underwater Challenges

Underwater visison is linked with adverse conditions that diminish image quality. Light refraction and scattering induces distortion and reduced visibility in the images, effectively making the segmentation task harder. Several methods have been used to mitigate the damage and restore quality to the images, caused by the subsea environment [60].

3.2.1 Localized Cropping for Image Segmentation

Data augmentation is a widely used technique that involves applying various transformations to the existing training data to create additional synthetic examples, increasing the size and diversity of the dataset. Due to the dataset being limited in terms of size, numerous data augmentation techniques were employed to artificially increase the size of the dataset, offline data augmentation refers to the process of pre-generating augmented versions of the training data before the training phase begins, while online data augmentation refers to the process of performing data augmentation on the fly during the training process. Generating offline data can be benefitial due to to

increasing the size of the dataset, ensuring higher diversity and more training data, however it may also lead to overfit if the generated data is too similar to the original data.

For this specific dataset, a custom transformation called *EdgeCrop* was applied that searched for zones in the border between regions of marine growth and the background and performed a crop in that region as displayed in figure 3.5. The goal of this transformation was to artificially increase the dataset while simultaneously maintaing its intrinsic properties due to all the synthetic data being generated from data inside the dataset. This method expanded the dataset from 150 images to **876 total images**, a **5.84x** increase. With this increase in data the new dataset has the following distribution:

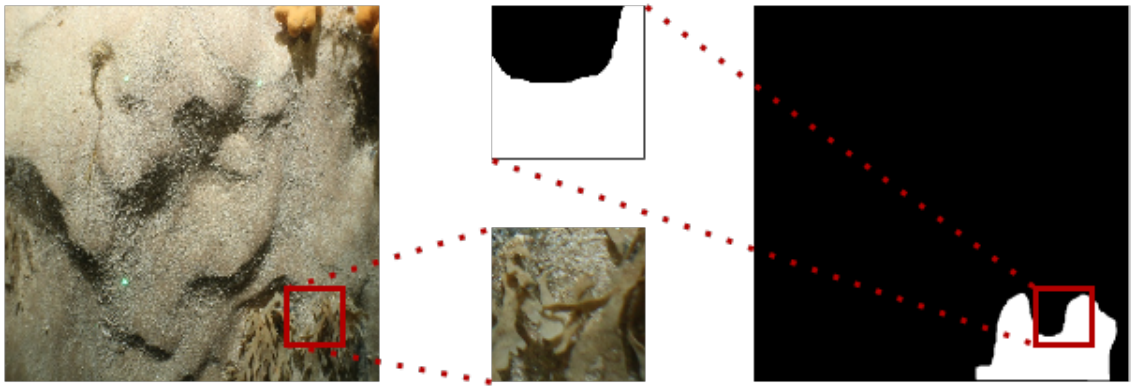

Figure 3.5: Custom transformation generating a new image-mask pair.

- **Class 0** (background) has 2.68×10^9 pixels, accounting for 93.38% of the total number of pixels in the dataset
- **Class 1** (marine growth) has 1.90×10^9 pixels, accounting for 6.62% of the total number of pixels in the dataset
- The image with the highest marine growth coverage has 97.63% of its total pixels covered by marine growth

The small increase in class 1 distribution is due to the generated images being stored in memory as 224x224 or 512x512 crops, depending on what input size the models are utilizing, **357x** or **68x** smaller than the original images with their original resolution of 5184x3456 that account for the majority of pixels in the dataset. Images are resized on-the-fly before feing fed to the model due to the computational resources not having enough memory to store images in their full resolution for training. For this work, the resizing has to be done in order to be able to train, however, other approaches may utilize the resize as a way to speed up training, due to the models having to process less information. When all images are resized the dataset has the following class distribution:

- **Class 0** (background) has 1.78×10^8 pixels, accounting for 77.37% of the total number of pixels in the dataset
- **Class 1** (marine growth) has 5.20×10^7 pixels, accounting for 22.63% of the total number of pixels in the dataset
- The image with the highest marine growth coverage has 97.63% of its total pixels covered by marine growth

The *EdgeCrop* transformation effectively creates an entirely new *Expanded Dataset*, with approximately 6x the amount of images and a more balanced distribution of classes.

3.2.2 On-the-fly Data Augmentation

Additional transformations to the data were made when the training data is directly fed into the model during training, with the transformations being applied to each sample in real-time. The transformations are typically random and vary from sample to sample, ensuring diversity in the augmented data presented to the model. The following data augmentation techniques were utilized:

- **Vertical Flip**: Performing a vertical flip with a probability of 50%,
- **Horizontal Flip**: Performing a horizontal flip with a probability of 50%,
- **Random Rotation**: Performing a rotation between -180 degrees and 180 degress with a probability of 100%,
- **Brightness Adjustment**: Adjust the brighness of the image between 0.75 and 1.25 of the original image brightness, with a probability of 100%. This was the only color transformation applied and as such its' values where chosen in order to introduce variability to the dataset without excessively altering the objects of interest that are being predicted.

3.3 Learning-based architecture for Marine Growth Segmentation

The success of training a high-performing image segmentation model is dependent on the appropriate selection and fine-tuning of various training parameters, which is thoroughly explored in this section. Following the literature review of chapter 2, several different models were developed:

- **U-Net + ResNet**: this model was based on the U-Net architecture featuring a ResNet50 backbone, a popular network in the image segmentation field. The weights were pre-trained on the ImageNet dataset to provide a better starting point for training

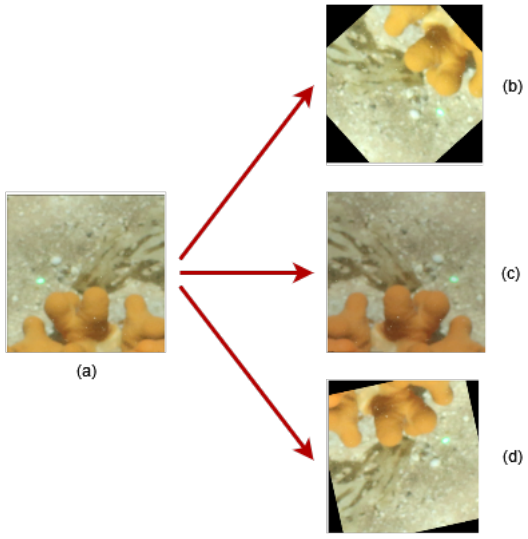

Figure 3.6: Sample image (a) generating 3 different images, (b), (c) and (d) via online data augmentation.

- **U-Net + VGG**: this model was based on the U-Net architecture featuring a VGG16 backbone, with weights pre-trained on the ImageNet dataset
- **DeeplabV3**: this model was developed following the architecture previously reviewed and was chosen due to having high performance on underwater segmentation and being the state-of-the-art in image segmentation [45]

Initially the models were given images resized to 224x224 to establish a baseline, after that the models were given images resized to 512x512 in order to evalute their performance on images with decreased loss of resolution. The analysis on the image sizes' impact on performance is explored in-depth in chapter 4. The loss function serves as a crucial component in the optimization process during model training. It quantifies the discrepancy between the predicted segmentation and the ground truth, providing a measure of how well the model is performing. By minimizing the loss, the model adjusts its parameters to improve the accuracy of the segmentation results. The loss function used in this work was the Dice coefficient loss. It measures the similarity between the predicted segmentation mask and the ground truth mask by computing the overlap between the two masks. The Dice loss DC_{loss} is derived from the Dice coefficient DC in equation 3.2, which is calculated as twice the intersection of the masks divided by the sum of their sizes in equation 3.1.

$$DC = \frac{2 \times |A \cap B|}{A \cup B} \tag{3.1}$$

$$DC_{loss} = 1 - DC \tag{3.2}$$

By utilizing the Dice loss, the model is encouraged to produce accurate and precise segmentation results, as it aims to maximize the overlap between the predicted and ground truth masks. The $2 \times |A \cap B|$ on the numerator emphasizes the importance of correctly identifying positive predictions, essentially encouraging the models to get correct predictions instead of avoiding wrong predictions. Since it is a differentiable loss function, it allows for gradient-based optimization during the training of deep learning models.

Regarding the hyperparameters, there are 3 choices to be made: the batch size, the number of epochs and the learning rate. The number of epochs is the number of complete passes through the entire dataset, it needs to be provide a balance between overfitting and underfitting, allowing the model to update long enough for it to reach high performance while simultaneously ending the training when the model starts to overfit. For this work the **number of epochs** chosen was **50**. The batch size refers to the number of samples processed before updating the model's weights, bigger batch sizes imply faster training times because their processing is done in parallel and sometimes better performance due to less sample noise, however the system needs more memory to process all the samples in parallel. For this work the **batch size** chosen was **3**. Lastly, the learning rate determines the step size the optimizer takes when updating the weights, increased learning rates increase convergence speed but may lead to an overshoot response resulting in a system unable to converge. For this work the chosen **learning rate** was $\mathbf{10^{-3}}$. The optimizer chosen was the **Adam** optimizer [64] due to its popularity in segmentation tasks. The idea behind the Adam optimizer is to adaptively adjust the learning rate for each parameter based on its historical gradients. This adaptive learning rate helps the optimizer converge faster and more reliably, especially when dealing with large-scale, high-dimensional problems. The weights can be updated using the following equation:

$$w_{t+1} = w_t - \frac{l_r \hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon_1} \tag{3.3}$$

Chapter 4

Experimental Results

The following chapter encompasses an in-depth examination of the results acquired through the experiments detailed in the preceding chapter. It initiates with a comprehensive analysis of the performance metrics employed, elucidating their significance and relevance. Subsequently, a meticulous assessment is conducted, encompassing both quantitative and qualitative analyses across diverse models and varying training conditions. Lastly, the chapter concludes by providing a comprehensive discussion and interpretation of the obtained results, addressing their implications and potential implications within the scope of the research.

4.1 Experimental setup

Evaluation metrics play a crucial role in assessing the performance and effectiveness of image segmentation algorithms. These metrics provide quantitative measures to evaluate how well the segmented regions align with the ground truth annotations or the desired segmentation masks. In this section, we will discuss some commonly used evaluation metrics for image segmentation. In binary classification tasks, labels can be of 2 types: positive with the value **1**, in this work's the positive value refers to pixels belonging to regions of marine growth as previously mentioned, and negative with the value **0**, referring to pixels belonging to regions of background or non-marine growth regions.

Models' predictions and the ground truth can be compared in a *confusion matrix* as displayed in figure 4.1. Models predictions in binary segmentation can be of 4 types:

- **True Positive (TP)**, when the ground truth label is positive and the model prediction is positive,
- **True Negative (TN)**, when the ground truth label is negative and the model prediction is negative,
- **False Positive (FP)**, when the ground truth label is negative and the model prediction is positive,

- **False Negative (FN)**, when the ground truth label is positive and the model prediction is negative.

		Predicted values	
		1	0
Actual values	1	TP	FN
	0	FP	TN

Figure 4.1: Confusion matrix

Using these types of predictions several metrics can be developed in order to evaluate models:

- **Pixel Accuracy**: The first intuitive metric to evaluate segmentation models is the pixel accuracy. This metric is calculated using the following formula:

$$PA = \frac{TP + TN}{TP + TN + FN + FP} \tag{4.1}$$

Pixel accuracy is calculated in equation 4.1 by dividing the amount of correct predictions $(TP + TN)$ by the total amount of predictions $(TP + TN + FN + FP)$. While it is one of the most popular metrics for image classification tasks, it has usually has some *class imbalance* problems in segmentation tasks. This is because classes are usually not evenly distributed in images. Using a sample segmentation mask (image (a) in figure 4.2) from the original dataset described in the previous chapter as an example, with a marine growth coverage percentage of 2.31%, a model predicting the image (b) 4.2 would have 97.69% pixel accuracy which would seem a good prediction but in reality when looking at both images it is apparent it is not an appropriate prediction. This is due to the amount of marine growth being just 2.31% of the total image and pixel accuracy taking into account the TN when calculating the metric. As a result of this work's dataset being heavily imbalanced as previously described, this metric will not be used.

- **Intersection over Union (IoU)**: This metric, also reffered to by *Jaccard Index*, measures the overlap between the predicted segmentation and the ground truth. It is calculated using the following formula:

Figure 4.2: Sample segmentation mask (a) and dummy prediction (b)

$$IoU = \frac{TP}{TP+FN+FP} \tag{4.2}$$

It calculates the area of overlap of the predicted segmentation and the ground truth, divided by the area of union between the ground gruth and the prediction segmentation. A higher IoU score indicates a better match between the predicted and ground truth segmentations. This metric was used extensively during this work to evaluate all the models developed.

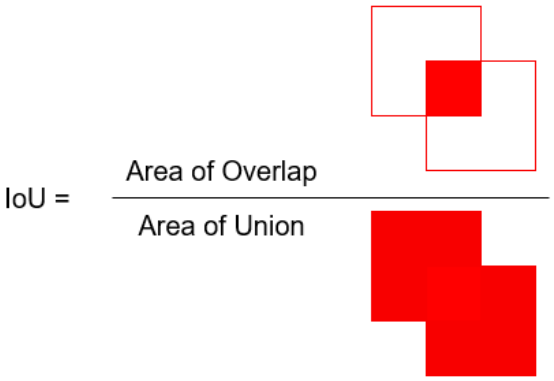

Figure 4.3: IoU visual example.

- **Dice coefficient**: This metric, often reffered to by *F1-score*, is a widely used evaluation metric in image segmentation tasks as previously mentioned. It measures the similarity or overlap between the predicted segmentation and the ground truth. The Dice coefficient is calculated as twice the intersection between the predicted and ground truth regions divided by the sum of their sizes (equation 4.3):

$$DC = \frac{2TP}{2TP+FN+FP} \tag{4.3}$$

To evaluate the models according to these metrics and also train the models the PyTorch framework with version 1.13.1 was used, due to its popularity among the community, ease of use and available support online. The system used to train is equipped with an Intel HM175 chipset and GeForce GTX 1060 with 8GB DDR4 for faster training on the GPU.

4.2 Marine Growth Segmentation

4.2.1 Initial Dataset Experiments

Initially models were developed and trained on the original dataset of 150 images. This was done to establish a baseline to define a point of reference for further improvents on the models and/or data. In the context of these experiments 4 models were developed, $VGG16_{224}$, $VGG16_{512}$, $DeeplabV3_{512}$ and $ResNet_{512}$.

The obtained results from the developed models are presented and analyzed in this section, shedding light on their performance in segmentation. The evaluation metrics used to assess the models' effectiveness are the DC loss and the IoU. The graphs in figure 4.4, accompanied by table 4.1 conclude that the models reach DC loss of around 0.6 and test IoU of 0.35. This means that the overlap between model prediction and ground truth is approximately 35%. The $ResNet_{512}$ is the most overfitted model due to having high discrepancy of performance in the train and test sets, which can be a result of a small dataset or low variability. It was expected that the $VGG16_{224}$ would be outperformed by the other models due to having to perform segmentation on images with a resolution of 224x224 that have a higher degree of loss of quality due to resizing than 512x512 images. However, this was not the case and the model achieves comparable performance with the other models.

Table 4.1: Quantitative performance on the initial dataset, better performance is characterized by higher IoUs and lower DC Losses.

Model	Train IoU	Test IoU	Train DC Loss	Test DC Loss
$VGG16_{224}$	0.303	0.347	0.604	0.592
$VGG16_{512}$	0.415	0.331	0.473	0.589
$DeeplabV3_{512}$	0.376	**0.365**	0.511	**0.564**
$ResNet_{512}$	**0.514**	0.238	**0.330**	0.665

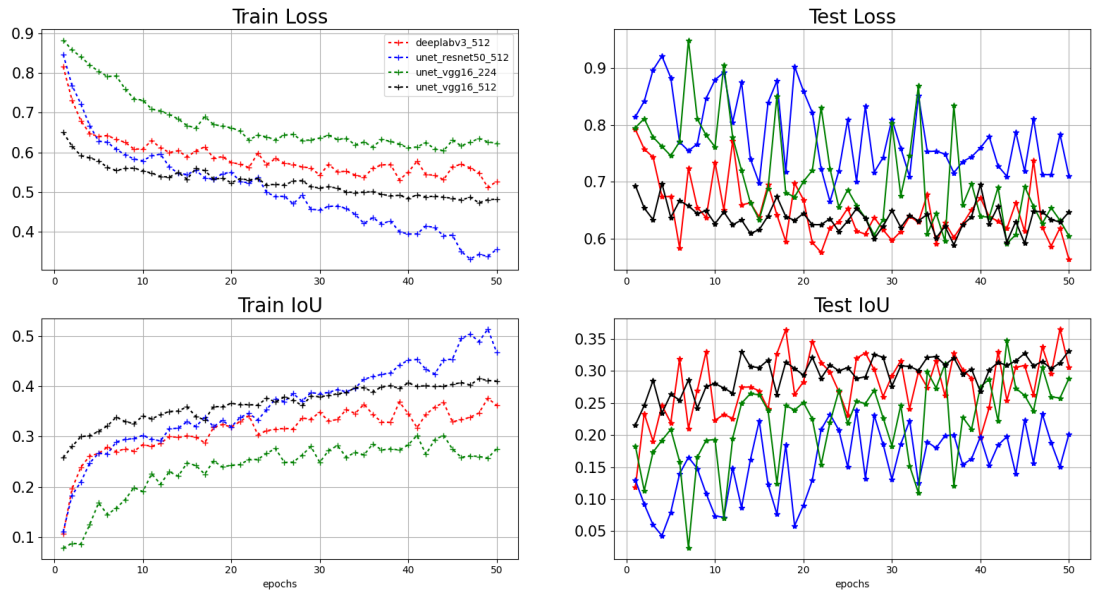

Figure 4.4: DC Loss (top) and IoU curves (bottom) for the train (left) and test (right) sets of the initial dataset. In blue $ResNet_{512}$, in red $DeeplabV3_{512}$, in green $VGG16_{224}$ and in black $VGG16_{512}$.

Overall these results show that the models are not achieving high performance on segmentation, to understand why this is happening a visual analysis of the models' predictions was made on figure 4.5 that contains 6 samples from the original test dataset and each models' prediction alongside the original image and ground truth. It can be concluded that regions of marine growth that have a bigger size such as the one depicted in the sample on rows 4 and 6 are more easily detected by the models with almost all the models detecting these shapes, at least partially. Smaller sized regions on the other hand, such as the ones displayed in rows 1, 3 and 5, are harder for the models to detect and almost all models provide mostly inaccurate predictions. The visual analysis suggests that the models' have more difficulty identifying smaller regions of marine growth, which is, to a certain point expected because these are zones that are more complicated to identify. However, the fact that the dataset is mostly comprised of images with small shapes of marine growth scattered through the image and only approximately 5% of the total pixels being marine growth can explain the poor segmentation results. With these issues in mind, it can be concluded that increasing the distribution of marine growth in the dataset can be beneficial for training these models.

4.2.2 Expanded Dataset Experiments

This section covers the performance of the models on the expanded dataset. Addressing the issues in the previous section, the dataset was expanded utilizing the EdgeCrop function described in the previous chapter. It is expected that this section describes better results than the previous section given that the dataset is approximately 6x bigger and the distribution of marine growth is more balanced.

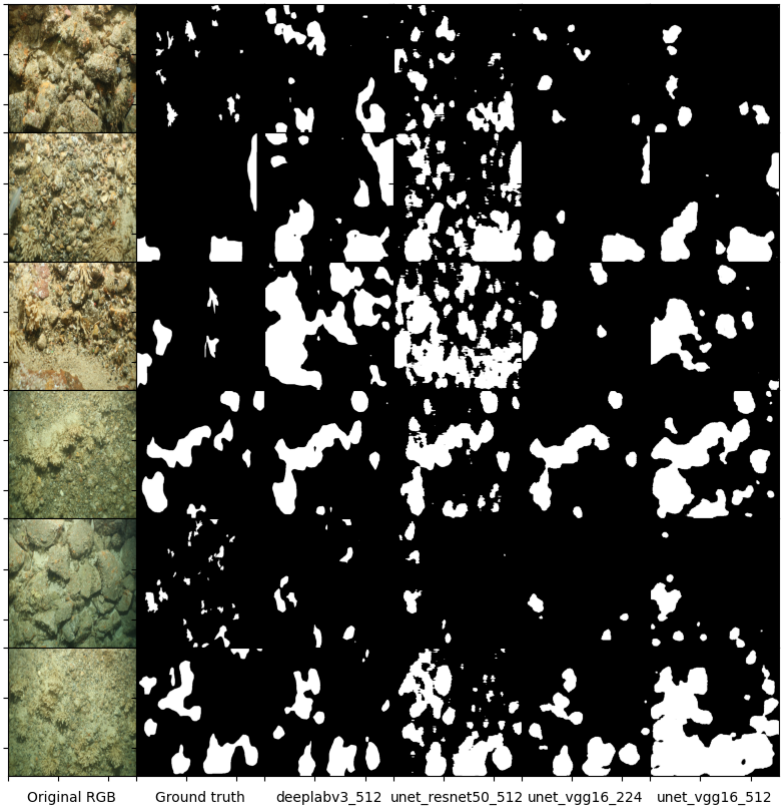

Figure 4.5: Qualitative analysis of 6 samples from the original dataset. Columns from left to right are original RGB, ground truth and $DeeplabV3_{512}$, $ResNet_{512}$, $VGG16_{224}$, $VGG16_{512}$ predictions.

The table 4.2 describes the performance of the developed models on the expanded dataset. Analyzing the figure 4.6 and table 4.2 and comparing them with the ones on the previous section it can be concluded that with the exception of the $VGG16_{224}$ model that attained the best performance across all metrics, the performance is approximately the same with higher degrees of overfit. The best model trained on this dataset exhibits a DC loss approximately 10% lower and a test IoU approximately 7% higher than the best model results on the previous sections, which can be explained by the increase in the amount of training data. However, these models present more overfit due to the increase in train IoU and train DC loss in relation to the previous section performance, but approximately the same performance on the validation IoU and test loss.

Table 4.2: Quantitative results for the expanded dataset better performance is characterized by higher IoUs and lower DC Losses.

Model	Train IoU	Test IoU	Train Loss	Test Loss
$VGG16_{224}$	**0.452**	**0.389**	**0.441**	**0.508**
$VGG16_{512}$	0.415	0.331	0.473	0.589
$DeeplabV3_{512}$	0.417	0.342	0.469	0.560
$ResNet_{512}$	0.439	0.324	0.445	0.590

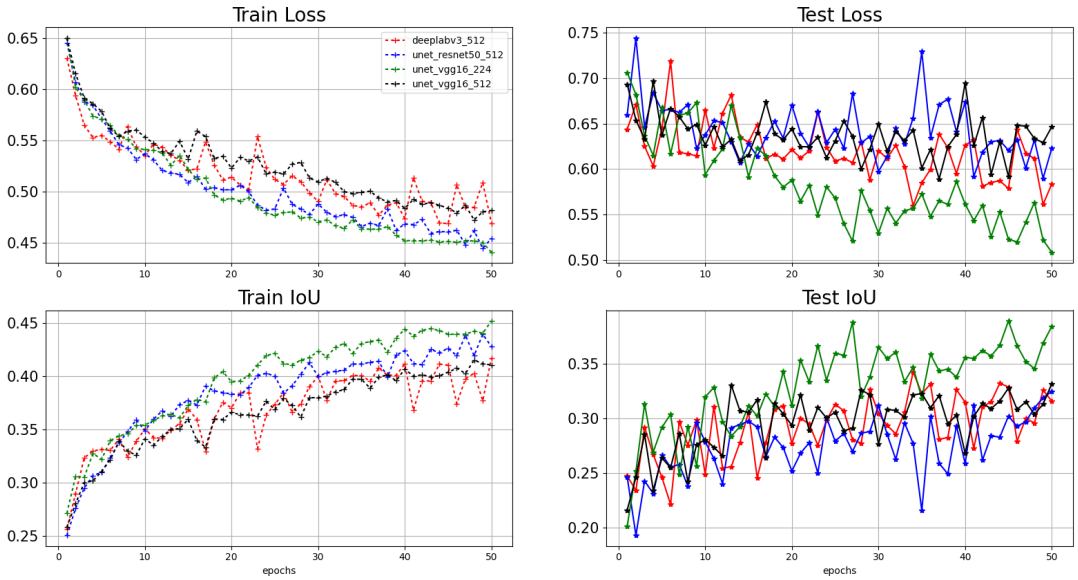

Figure 4.6: DC Loss (top) and IoU curves (bottom) for the train (left) and test (right) sets of the expanded dataset. In blue $ResNet_{512}$, in red $DeeplabV3_{512}$, in green $VGG16_{224}$ and in black $VGG16_{512}$.

The experimental analysis conducted on the expanded dataset revealed contrasting results between the visual assessment and numerical evaluation of the segmentation predictions, as depicted in Figure 4.7. The augmentation technique employed, known as EdgeCrop, involved augmenting the dataset by introducing localized crops focusing on the border regions between the foreground and background. While this approach successfully increased the dataset size and improved class balance, it also led to a predominant inclusion of images with reduced visual context compared to the original images, due to their cropped nature with a fraction of the original image's dimensions. To ensure consistency, the same samples were utilized for comparing the visual analysis of the models trained on the expanded dataset with those trained on the original dataset. However, it is important to note that the models trained on the expanded dataset were exposed to significantly reduced visual context and higher class distribution due to the abundance of cropped images in the dataset, which can explain worse performance on samples present on the original dataset.

4.3 Testing in Real World Scenario

Due to the models exhibiting a degree of overfit, further testing was done to evaluate the generalization capabilities on entirely different data. The data acquired are images containing marine growth surrounding underwater structures [65]. Figure 4.8 displays some predictions the best performing model made on this data. Since the data didn't have segmentation masks, numerical metrics, such as DC loss and IoU were not able to be calculated, leaving only the option of visual analysis of the segmentation results. In most samples the model either, fails to predict anything or

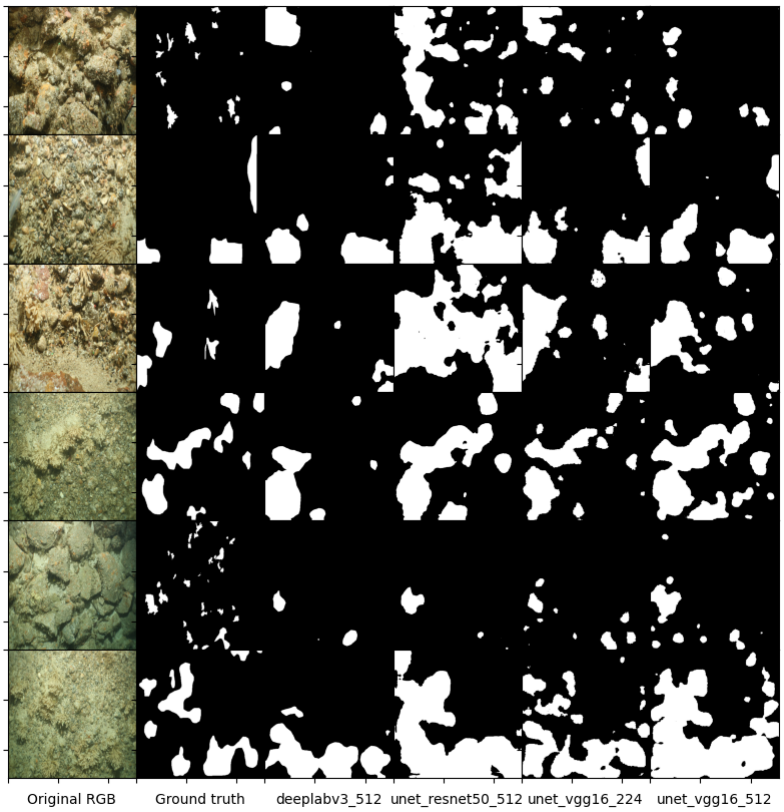

Figure 4.7: Qualitative analysis of 6 samples from the expanded dataset. Columns from left to right are original RGB, ground truth and $DeeplabV3_{512}$, $ResNet_{512}$, $VGG16_{224}$, $VGG16_{512}$ predictions.

predicts incorrectly; this can be explained in different ways. The first one is that the model lacks generalization capabilities, indicating it is most likely overfit to this work's dataset. Another one could be the lighting conditions, as this new dataset features very dark images and the models are trained on images with better lighting conditions. Finally, this work's dataset species of marine growth may not be the same ones present in the new data, which the model isn't trained to identify.

4.4 Conclusion

The original dataset used in this study is characterized by its small scale and contains the presence of marine growth regions with intricate shapes. These shapes suffer a loss in quality when resized down to the desired dimensions, posing a challenge for the models to learn accurate predictions. Consequently, the models exhibited difficulty in consistently predicting these shapes, showing limited capability to detect smaller-sized regions of marine growth with detailed and irregular boundaries. However, data augmentation techniques were employed as a mitigation strategy to address these issues. The augmentation process involved generating new images that specifically emphasized the borders between marine growth regions and the background, thereby augmenting the dataset and balancing the distribution of marine growth instances. It was observed that only

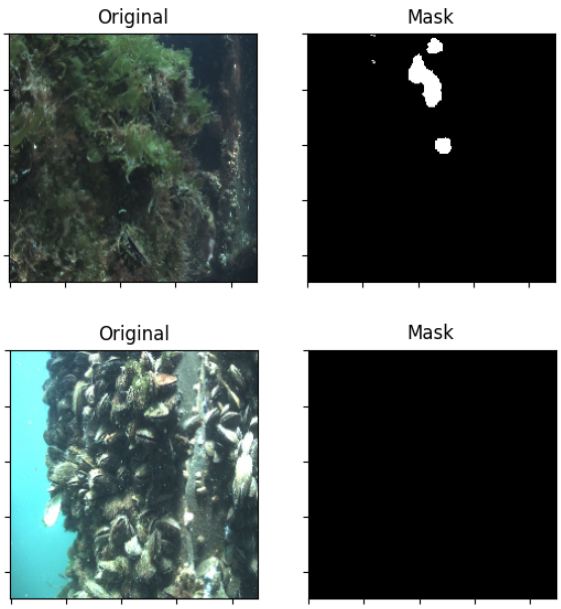

Figure 4.8: Visual analysis on new data.

one model, namely $VGG16_{224}$, demonstrated improved performance with the application of data augmentation, while other models showed an increased degree of overfitting.

Considering that the selected models are state-of-the-art in image segmentation and have exhibited higher performance in underwater segmentation tasks, as discussed in Chapter 2, these findings suggest that further improvement in this specific task primarily relies on enhancing the dataset quality and diversity.

Chapter 5

Conclusion and Future Work

This work explores the underwater vision field utilizing state-of-the-art deep learning algorithms. It proposes to identify regions of marine growth within underwater images to facilitate maintenance processes in offshore structures. The main obstacles to overcome in this work were the negative effects that the underwater environment has on images and the lack of data. To surpass these challenges, several methods were studied and the EdgeCrop transformation was developed that searches for zones between foreground and background and generates new data by cropping the original image in that zone. This effectively enlarged the dataset approximately 6x and provided a more balanced distribution of classes.

In regards to model performance, $DeeplabV3_{512}$ was the best model trained on the original dataset, having achieved test DC loss of 0.564 and test IoU of 0.365. $ResNet_{512}$ is showing signals of being overfit due to displaying high performance metrics on the train set, achieving train IoU and train DC loss of 0.514 and 0.330, but having the lowest performance on the test set of all the 4 models with 0.238 IoU and 0.665 DC loss. On the expanded dataset the model $VGG16_{224}$ achieved the best segmentation metrics across both train and test set, with DC test loss of 0.508 and test IoU of 0.389, the DC loss obtained with this model is 10% lower and the test IoU 4% higher than the best results achieved in the original dataset. This model, however, demonstrated a degree of overfitting due to being unable to identify marine growth in an entirely new dataset.

To take this research to the next step, a promising approach would be to expand the original dataset. The dataset contains high resolution images that don't display significant damage by the underwater environment, however, the small size of the dataset coupled with only 5% of it being marine growth doesn't enable the models to effectively learn the complex patters that represent the objects being identified. Additionally, due to the complex and detailed shapes of the objects being predicted, an increase on the computational resources available may prove beneficial by decreasing the resize from the original resolution to the resolution of images being fed to the model. Overall, this work makes significant advancements in building the infrastructure and baselines for a model capable of performing segmentation on marine growth, however, there is significant room for improvement, specially on the data used to train the models.

References

[1] Xiaoqing Liu, Kunlun Gao, Bo Liu, Chengwei Pan, Kongming Liang, Lifeng Yan, Jiechao Ma, Fujin He, Shu Zhang, Siyuan Pan, and Yizhou Yu. Advances in deep learning-based medical image analysis. *Health Data Science*, 2021, 2021. doi:10.34133/2021/8786793.

[2] Daniel Filipe Campos, Maria Pereira, Aníbal Matos, and Andry Maykol Pinto. Diius - distributed perception for inspection of aquatic structures. In *OCEANS 2021: San Diego – Porto*, pages 1–5, 2021. doi:10.23919/OCEANS44145.2021.9705939.

[3] Maria Inês Pereira, Rafael Marques Claro, Pedro Nuno Leite, and Andry Maykol Pinto. Advancing autonomous surface vehicles: A 3d perception system for the recognition and assessment of docking-based structures. *IEEE Access*, 9:53030–53045, 2021. doi:10.1109/ACCESS.2021.3070694.

[4] Alvin Sarraga Alon, Jonel Macalisang, Ryan Carreon Reyes, Rovenson V. Sevilla, and Gemma D. Belga. Watercraft-net: A deep inference vision approach of watercraft detection for maritime surveillance system using optical aerial images. In *2020 IEEE 7th International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, pages 1–5, 2020. doi:10.1109/ICETAS51660.2020.9484279.

[5] Andry Maykol Pinto, João V. Amorim Marques, Daniel Filipe Campos, Nuno Abreu, Aníbal Matos, Martio Jussi, Robin Berglund, Jari Halme, Petri Tikka, João Formiga, Christian Verrecchia, Serena Langiano, Clara Santos, Nuno Sá, Jaap-Jan Stoker, Fabrice Calderoni, Shashank Govindaraj, Alexandru But, Leslie Gale, David Ribas, Natalia Hurtós, Eduard Vidal, Pere Ridao, Patryk Chieslak, Narcis Palomeras, Stefano Barberis, and Luca Aceto. Atlantis - the atlantic testing platform for maritime robotics. In *OCEANS 2021: San Diego – Porto*, pages 1–5, 2021. doi:10.23919/OCEANS44145.2021.9706059.

[6] Daniel Campos, Aníbal Matos, and Andry Pinto. Multi-domain inspection of offshore wind farms using an autonomous surface vehicle. *SN Applied Sciences*, 3, 04 2021. doi:10.1007/s42452-021-04451-5.

[7] Pedro Leite and Andry Pinto. Fusing heterogeneous tri-dimensional information for reconstructing submerged structures in harsh sub-sea environments. 01 2023. doi:10.2139/ssrn.4409685.

[8] Andry Maykol Pinto and Anibal C. Matos. Maresye: A hybrid imaging system for underwater robotic applications. *Information Fusion*, 55:16–29, 2020. doi:10.1016/j.inffus.2019.07.014.

[9] Sidum Adumene and Hope Ikue-John. Offshore system safety and operational challenges in harsh arctic operations. *Journal of Safety Science and Resilience*, 3(2):153–168, 2022. doi:https://doi.org/10.1016/j.jnlssr.2022.02.001.

[10] Nikolaos Skliris, Robert Marsh, Meric Srokosz, Yevgeny Aksenov, Stefanie Rynders, and Nicolas Fournier. Assessing extreme environmental loads on offshore structures in the north sea from high-resolution ocean currents, waves and wind forecasting. *Journal of Marine Science and Engineering*, 9(10), 2021. doi:10.3390/jmse9101052.

[11] Moacir Apolinario and Ricardo Coutinho. *Understanding the biofouling of offshore and deep-sea structures*, pages 132–147. 05 2009. doi:10.1533/9781845696313.1.132.

[12] Renato Silva, Aníbal Matos, and Andry Pinto. Multi-criteria metric to evaluate motion planners for underwater intervention. *Autonomous Robots*, 46:1–13, 09 2022. doi:10.1007/s10514-022-10060-x.

[13] Maria Inês Pereira, Pedro Nuno Leite, and Andry Maykol Pinto. A 3-d lightweight convolutional neural network for detecting docking structures in cluttered environments. *Marine Technology Society Journal*, 2021. doi:10.4031/MTSJ.55.4.9.

[14] Satja Sivcev, Joseph Coleman, Edin Omerdic, Gerard Dooly, and Daniel Toal. Underwater manipulators: A review. *Ocean Engineering*, 163:431–450, 09 2018. doi:10.1016/j.oceaneng.2018.06.018.

[15] Simon Pedersen, Jesper Liniger, Fredrik F. Sørensen, and Malte von Benzon. On marine growth removal on offshore structures. In *OCEANS 2022 - Chennai*, pages 1–6, 2022. doi:10.1109/OCEANSChennai45887.2022.9775498.

[16] Malte von Benzon, Fredrik Sørensen, Jesper Liniger, Simon Pedersen, Sigurd Klemmensen, and Kenneth Schmidt. Integral sliding mode control for a marine growth removing rov with water jet disturbance. In *2021 European Control Conference (ECC)*, pages 2265–2270, 2021. doi:10.23919/ECC54610.2021.9655050.

[17] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014. doi:10.1109/CVPR.2014.220.

[18] F.H.Y. Chan, F.K. Lam, and Hui Zhu. Adaptive thresholding by variational method. *IEEE Transactions on Image Processing*, 7(3):468–473, 1998. doi:10.1109/83.661196.

[19] F. Wong, R. Nagarajan, S. Yaacob, A. Chekima, and N.-E. Belkhamza. An image segmentation method using fuzzy-based threshold. In *Proceedings of the Sixth International Symposium on Signal Processing and its Applications (Cat.No.01EX467)*, volume 1, pages 144–147 vol.1, 2001. doi:10.1109/ISSPA.2001.949796.

[20] Nobuyuki Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979. doi:10.1109/TSMC.1979.4310076.

[21] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theory*, 28(2):129–136, 1982. doi:10.1109/TIT.1982.1056489.

[22] Jinyong Cheng, Ruojuan Xue, Wenpeng Lu, and Ruixiang Jia. Segmentation of medical images with canny operator and gvf snake model. In *2008 7th World Congress on Intelligent Control and Automation*, pages 1777–1780, 2008. doi:10.1109/WCICA.2008.4593191.

[23] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. doi:10.1109/TPAMI.1986.4767851.

[24] M. Mary Synthuja Jain Preetha, L. Padma Suresh, and M. John Bosco. Image segmentation using seeded region growing. In *2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*, pages 576–583, 2012. doi:10.1109/ICCEET.2012.6203897.

[25] B. Lakshmipriya, K. Jayanthi, Biju Pottakkat, and G. Ramkumar. Liver segmentation using bidirectional region growing with edge enhancement in nsct domain. In *2018 ieee international conference on system, computation, automation and networking (icscan)*, pages 1–5, 2018. doi:10.1109/ICSCAN.2018.8541257.

[26] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994. doi:10.1109/34.295913.

[27] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. doi:10.1038/nature14539.

[28] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998. doi:10.1109/5.726791.

[29] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. doi:10.48550/arXiv.1511.08458.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. doi:10.1145/3065386.

[31] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. doi:10.1109/CVPR.2009.5206848.

[32] Rikiya Yamashita, Mizuho Nishio, Richard K. G. Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9:611 – 629, 2018. doi:10.1007/s13244-018-0639-9.

[33] Muhamad Yani, S Irawan, and Casi Setianingsih. Application of transfer learning using convolutional neural network method for early detection of terry's nail. *Journal of Physics: Conference Series*, 1201:012052, 05 2019. doi:10.1088/1742-6596/1201/1/012052.

[34] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986. doi:10.1038/323533a0.

[35] Xue Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168:022022, 02 2019. doi:10.1088/1742-6596/1168/2/022022.

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. doi:10.48550/arXiv.1505.04597.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. doi:10.1109/CVPR.2016.90.

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. doi:10.48550/arXiv.1409.1556.

[39] Zhi-Peng Jiang, Yi-Yang Liu, Zhen-En Shao, and Ko-Wei Huang. An improved vgg16 model for pneumonia image classification. *Applied Sciences*, 11(23), 2021. doi:10.3390/app112311185.

[40] Sunitha Basodi, Chunyan Ji, Haiping Zhang, and Yi Pan. Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, 3(3):196–207, 2020. doi:10.26599/BDMA.2020.9020004.

[41] Souvik Kundu, Hesham Mostafa, Sharath Nittur Sridhar, and Sairam Sundaresan. Attention-based image upsampling, 2020. doi:10.48550/arXiv.2012.09904.

[42] Pedro Nuno Leite and Andry Maykol Pinto. Exploiting motion perception in depth estimation through a lightweight convolutional neural network. *IEEE Access*, 9:76056–76068, 2021. doi:10.1109/ACCESS.2021.3082697.

[43] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation, 2016. doi:10.1109/TPAMI.2016.2644615.

[44] Pedro Vianna, Ricardo Farias, and Wagner Pereira. U-net and segnet performances on lesion segmentation of breast ultrasonography images. *Research on Biomedical Engineering*, 37, 03 2021. doi:10.1007/s42600-021-00137-4.

[45] Md Jahidul Islam, Chelsey Edge, Yuyang Xiao, Peigen Luo, Muntaqim Mehtaz, Christopher Morse, Sadman Sakib Enan, and Junaed Sattar. Semantic segmentation of underwater imagery: Dataset and benchmark, 2020. doi:10.1109/IROS45743.2020.9340821.

[46] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. doi:10.48550/arXiv.1706.05587.

[47] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2016. doi:10.48550/arXiv.1412.7062.

[48] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017. doi:10.1109/TPAMI.2017.2699184.

[49] Yifan Si, Dawei Gong, Yang Guo, Xinhua Zhu, Qiangsheng Huang, Julian Evans, Sailing He, and Yaoran Sun. An advanced spectral–spatial classification framework for hyperspectral imagery based on deeplab v3+. *Applied Sciences*, 11:5703, 06 2021. doi:10.3390/app11125703.

[50] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. doi:10.1007/s11263-009-0275-4.

[51] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding, 2016. doi:10.1109/CVPR.2016.350.

[52] António Pedro Oliva Afonso and Andry Maykol Pinto. Underwater object recognition: A domain-adaption methodology of machine learning classifiers. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–6, 2019. doi:10.23919/OCEANS40490.2019.8962693.

[53] Diogo Ferreira Duarte, Maria Inês Pereira, and Andry Maykol Pinto. Multiple vessel detection and tracking in harsh maritime environments. In *OCEANS 2021: San Diego – Porto*, pages 1–5, 2021. doi:10.23919/OCEANS44145.2021.9705954.

[54] Diogo Ferreira Duarte, Maria Inês Pereira, and Andry Maykol Pinto. Multiple vessel detection in harsh maritime environments. *Marine Technology Society*, 5:58–67, 2022. doi:10.4031/MTSJ.56.5.07.

[55] Daniel Filipe Campos, Aníbal Matos, and Andry Maykol Pinto. Modular multi-domain aware autonomous surface vehicle for inspection. *IEEE Access*, 10:113355–113375, 2022. doi:10.1109/ACCESS.2022.3217504.

[56] Daniel Filipe Campos, Aníbal Matos, and Andry Maykol Pinto. Multi-domain mapping for offshore asset inspection using an autonomous surface vehicle. In *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 221–226, 2020. doi:10.1109/ICARSC49921.2020.9096097.

[57] 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), 2022. doi:10.3390/jmse10020241.

[58] Paulo Drews-Jr, Erickson Nascimento, Silvia Botelho, and Mario Campos. Underwater depth estimation and image restoration based on single images. *IEEE Computer Graphics and Applications*, 36:24–35, 03 2016. doi:10.1109/MCG.2016.26.

[59] Jinkang Wang, Xiaohui He, Faming Shao, Guanlin Lu, Ruizhe Hu, and Qunyan Jiang. Semantic segmentation method of underwater images based on encoder-decoder architecture. *PLOS ONE*, 17(8):1–19, 08 2022. doi:10.1371/journal.pone.0272666.

[60] Yang Zhou, Jiangtao Wang, Baihua Li, Qinggang Meng, Emanuele Rocco, and Andrea Saiani. Underwater scene segmentation by deep neural network. pages 44–47, 01 2019. doi:10.31256/UKRAS19.12.

[61] Paulo Drews-Jr, Isadora Souza, Igor Maurell, Eglen Protas, and Silvia Botelho. Underwater image segmentation in the wild using deep learning. *Journal of the Brazilian Computer Society*, 27, 12 2021. doi:10.1186/s13173-021-00117-7.

[62] J.S. Jaffe. Computer modeling and the design of optimal underwater imaging systems. *IEEE Journal of Oceanic Engineering*, 15(2):101–111, 1990. doi:10.1109/48.50695.

[63] B. L. McGlamery. A Computer Model For Underwater Camera Systems. In Seibert Quimby Duntley, editor, *Ocean Optics VI*, volume 0208, pages 221 – 231. International Society for Optics and Photonics, SPIE, 1980. doi:10.1117/12.958279.

[64] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. doi:10.48550/arXiv.1412.6980.

[65] Joao Dionisio, Pedro Pereira, Pedro Leite, Joao Manuel Tavares, and Andry Pinto. Nereon - an underwater dataset for monocular depth estimation. In *OCEANS 2023: Limerick*, 2023.