Vision-based classification for underwater safety critical applications

Luiza Ribeiro Marnet

Advisors: Prof. Andrzej Wąsowski
Dr. Yury Brodskiy
Dr. Stella Grasshof
Submitted: December 2024

IT UNIVERSITY OF COPENHAGEN

"Entrega o teu caminho ao Senhor; confia nele, e ele tudo fará."
BIBLIA, 1977, Tradução de João Ferreira de Almeida

Abstract

Computer vision is crucial for ensuring a safe future for the quickly growing underwater infrastructure monitoring industry, which currently relies primarily on labour-intensive and costly manual methods. Over the last twelve years, deep learning models have revolutionized the field of computer vision and have been applied across various domains. These models could potentially assist in developing underwater surveys, for example, by analyzing videos and image data from equipment inspection and environment monitoring, thereby automating the process and reducing the time spent on visual inspections. Despite the success of deep learning models, underwater images impose challenges not faced with in-air images. Underwater images are generally of poor quality with issues such as blur, haziness, non-uniform illumination, and color degradation. These factors raise concerns about the performance of well-known deep learning models in underwater applications.

As a subset of machine learning, deep learning models are data-driven, and the quality of the training data impacts the model's performance. Furthermore, as a consequence of the large number of parameters, they are data-greedy, requiring large datasets for effective training. However, large underwater datasets are difficult to generate and not widely available. Collecting underwater data relies on the availability of underwater vehicles, specialists to perform the survey, and favorable weather and water conditions. Once collected, the data needs to be labeled. Data annotation is an extremely laborious task, prone to human errors, which reduces the quality of the final dataset.

Furthermore, a well-known problem in deep learning is the overconfidence of the models, which can predict outputs with high probability even for inputs out of distribution (OOD) of the training data. This dissertation leverages this overconfidence by employing predictive uncertainty to identify the most important images for labelling, addressing limited financial or human resources for data annotation.

An extensive review of the state-of-the-art of deep learning models applied to underwater images concluded that predictive uncertainty is rarely used in this field. The literature review also revealed the scarcity of large underwater datasets and the researchers' efforts to develop models and training strategies to overcome this limitation. This lack of publicly available datasets is even more pronounced for industrial applications. To address this gap, this thesis supported the development and release of three publicly available underwater RGB datasets: MIMIR (synthetic), SubPipe, and MarinaPipe (real-world datasets). The contribution for the MIMIR project consisted of evaluating the dataset usability in the context of pipeline segmentation. In the SubPipe and MarinaPipe projects, the work consisted on annotating and evaluating the datasets for image segmentation. MarinaPipe was recorded in very shallow waters, resulting in images with visible sunlight rays, and contains pipelines occluded by the sediments from the marina floor. SubPipe was recorded in deeper waters, leading to darker images, with many parts of the pipeline covered by sand. The links for downloading these three datasets are available in the REMARO GitHub (https://github.com/remaro-network).

The overconfidence in deep learning models is an overwhelming concern. However, it is possible to leverage this overconfidence by calculating the predictive uncertainty to assess the models' lack of knowledge and use this information to reduce the effort required to generate labeled datasets. This thesis investigated the hypotheses that, given a model trained on synthetic data, the predictive uncertainty enables the selection of real-world images about which the model demonstrates little to no knowledge, for fine-tuning the model and bridging the synthetic-to-reality gap that exists even for photorealistic images. Selecting images based on uncertainty, calculated with Monte Carlo dropout, resulted in a model with better performance compared to randomly selecting the same amount of images.

Additionally, this research explored using predictive uncertainty calculated with Monte Carlo dropout for active learning in the underwater domain. When training with real underwater pipeline images and using uncertainty-driven active learning for selecting images, at least 15.9% fewer annotations were needed to achieve the same performance as a model trained on randomly selected images. In addition, this PhD research trained a generalized few-shot segmentation model and used predictive uncertainty to evaluate the reliability of the predictions using mutual information and entropy.

The experiments performed indicate that predictive uncertainty both increases the reliability of deep learning models and optimizes the use of human and financial resources available for data annotation by selecting the most informative data samples.

Resumé*

Computer vision er afgørende for at sikre en sikker fremtid i den hurtigt voksende industri inden for overvågning af undervandsinfrastruktur, som i øjeblikket primært er afhængig af arbejdskrævende og kostbare manuelle metoder. Over de sidste tolv år har deep learning-modeller revolutioneret området inden for computer vision og er blevet anvendt på tværs af forskellige domæner. Disse modeller kan potentielt bistå med udvikling af undervandsundersøgelser, for eksempel ved at analysere videoer og billeddata fra udstyrsinspektioner og miljøovervågning hvorved processen kan automatiseres, og tiden brugt på visuelle inspektioner kan reduceres.På trods af deep learning-modellernes succes så repræsenterer undervandsbilleder udfordringer, der ikke ses ved billeder over vand. Undervandsbilleder er generelt af dårlig kvalitet med problemer som sløring, uklarhed, ujævn belysning og farvedegradering. Disse faktorer rejser bekymringer om ydeevnen af velkendte deep learning-modeller i undervands regi.

Som en underkategori af maskinlæring er deep learning-modeller datadrevne, og kvaliteten af træningsdata påvirker modellens præstation. Desuden kræver de på grund af det store antal parametre omfattende datasæt for effektiv træning. Store undervandsdatasæt er dog vanskelige at generere og ikke nemt tilgængelige. Indsamling af undervandsdata afhænger af tilgængeligheden af undervandsfartøjer, specialister til at udføre undersøgelsen samt gunstige vejr- og vandforhold. Når dataene er indsamlet, skal de annoteres. Dataannotering er en ekstremt arbejdskrævende opgave der er sårbar overfor menneskelige fejl, hvilket reducerer kvaliteten af det endelige datasæt.

Herudover, er et velkendt problem inden for deep learning, modellernes overkonfidens, som gør at de kan forudsige outputs med høj sandsynlighed, selv for inputs uden for distributionen (OOD) af træningsdataene. Denne afhandling udnytter prædiktiv usikkerhed til at identificere de vigtigste billeder til annotering og dermed addressere udfordringer med begrænsede økonomiske eller menneskelige ressourcer til dataannotering.

En omfattende gennemgang af state-of-the-art deep learning-modeller anvendt på undervandsbilleder konkluderede, at prædiktiv usikkerhed sjældent bruges i dette felt. Litteraturgennemgangen

*Text translated with ChatGPT [1] and revised by a native Danish speaker.

afslørede også manglen på store undervandsdatasæt samt forskernes bestræbelser på at udvikle modeller og træningsstrategier for at overkomme denne begrænsning. Denne mangel på offentligt tilgængelige datasæt er endnu mere udtalt for industrielle applikationer. For at tackle denne udfordring understøttede denne afhandling udviklingen og udgivelsen af tre undervands-RGB-datasæt: MIMIR (syntetisk), SubPipe og MarinaPipe (real-world datasæt). Bidraget til MIMIR-projektet bestod i at evaluere datasættets anvendelighed i forbindelse med segmentering af rørledninger. I SubPipe- og MarinaPipe-projekterne bestod arbejdet i at annotere og evaluere datasættene til billedsegmentering. MarinaPipe blev optaget på meget lavt vand, hvilket resulterede i billeder med synlige solstråler og rørledninger dækket af sedimenter fra marinaens bund. SubPipe blev optaget på dybere vand, hvilket førte til mørkere billeder med mange dele af rørledningen dækket af sand. Links til download af disse tre datasæt er tilgængelige på REMARO GitHub (https://github.com/remaro-network).

Overkonfidens i deep learning-modeller er en betydelig bekymring. Det er dog muligt at udnytte denne overkonfidens ved at beregne den prædiktive usikkerhed for at vurdere modellernes manglende viden og bruge denne information til at reducere indsatsen, der kræves for at generere annoterede datasæt. Denne afhandling undersøgte hypotesen om, at givet at en model trænes på syntetiske data, så kan prædiktiv usikkerhed muliggøre udvælgelse af de real-world billeder, som modellen har lidt eller ingen viden om, til at finjustere modellen og lukke hullet der eksisterer mellem rigtige og syntetiske billeder, selv for fotorealistiske billeder. Udvælgelse af billeder baseret på usikkerhed, beregnet med Monte Carlo dropout, resulterede i en model med bedre præstation sammenlignet med tilfældigt udvalgte billeder.

Herudover blev der i denne forskning undersøgt brugen af prædiktiv usikkerhed beregnet med Monte Carlo dropout til aktiv læring i undervandsdomænet. Ved træning med reelle undervandsbilleder af rørledninger og anvendelse af usikkerhedsbaseret aktiv læring til billedudvælgelse var der behov for mindst 15,9% færre annoteringer for at opnå samme ydeevne som en model trænet på tilfældigt udvalgte billeder. Endelig blev der i denne ph.d. trænet en few-shot segmenteringsmodel og anvendte prædiktiv usikkerhed til at evaluere pålideligheden af forudsigelser ved hjælp af mutual information og entropi. De udførte eksperimenter indikerer, at prædiktiv usikkerhed både øger pålideligheden af deep learning-modeller og optimerer brugen af de menneskelige og økonomiske ressourcer der er tilgængelig til dataannotering ved at udvælge de mest informative datasamples.

Acknowledgments

This dissertation is the result of years of dedication, learning, and an invaluable network of people and institutions — along with several sleepless nights.

First and foremost, I want to thank my parents, Moema and Robson, for guiding me since my (literal) first steps and for teaching me the value of knowledge, kindness, and fairness. Your encouragement and sacrifices provided me with every opportunity to grow, both academically and personally. To my brother, Filipe, thank you for reminding me of what truly matters. To the three of you, thank you for your unconditional love, for understanding the distance, and for supporting this wild idea of studying on the other side of the globe. To my grandparents, Walter, Hilda, Editor, and Mirtis — some of whom are no longer with us — thank you for teaching my parents the value of education and hard work, so they could pass those lessons on to me.

To my partner, Frederik, thank you for your patience, understanding, and unwavering support throughout this process. You held my hand and pushed me forward when I thought I had reached my limits. Thank you for understanding the late nights, missed weekends, and skipped holidays — and for all the times you drove or flew to visit me. Your support, whether cooking me dinner or simply being by my side, has been my anchor.

To my Brazilian friends and family, thank you for being my connection to home, for your encouragement, and for understanding my long absences. To my friends in Aarhus, thank you for warming up the cold Danish winters, and for brightening the rainy summers, with your company.

To my educators who shaped my academic foundation, I am forever grateful. To my dear professors and supervisors at the Fluminense Federal Institute - Robson Santos, Marcos Cruz e Nelson Jr. - thank you for inspiring my passion for coding and research. - To my mentors at the Federal University of Rio de Janeiro – Carlos Carlim and Luiz Pereira Calôba - I owe you my passion for machine learning.

To my REMARO colleagues, thank you for creating such a stimulating research environment. I feel fortunate to have been part of this amazing group, sharing weeks of training, research, and countless fun evenings across Europe, America, and Asia. To the SQUARE group, thank you for making me feel welcome, both remotely and in the office, and for the many inspiring meetings and conversations. To my former Aarhus University Lab, thank you for the months we shared an office space and for the fun lunch breaks together. To the team at OceanScan, thank you for hosting me for two months and teaching me about the challenges of collecting the underwater images, which I was eager to work with.

To my colleagues at EIVA, thank you for sharing your expertise, office space, datasets, and computational resources. I will never forget my warm welcome with a basket of Danish candies, the Christmas parties, and the board game nights. A special thanks to Ole Kristensen for the opportunity to be part of your team. Your kindness made a meaningful difference in my daily life. I also want to thank Antonio Felipe Silva for introducing me to this opportunity and for always sharing your excitement for research plans. I am also thankful to Marie Grønborg Bak, Anette Sponholtz, and Joakim Koed Nyekjær for their invaluable support, whether for bureaucratic, financial, or IT issues.

I am deeply grateful for the financial support provided by the Marie Skłodowska-Curie Fellowship, which made this research possible. I also thank ITU and Aarhus University for their institutional support and guidance throughout my thesis.

Finally, to my supervisors — Yury, Stella, and Andrzej — I owe an enormous debt of gratitude. Your expertise, patience, and guidance have been invaluable. I feel incredibly fortunate to have been your student. Thank you for all the knowledge you shared and for your support.

To all of you, thank you for your support and inspiration. I could not have done this without you. Each of you has been a fundamental pillar in this journey.

Recognition of funding and grants

This thesis was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 956200, REMARO.

List of Papers

Below is the list of papers developed during this PhD study:

1. **Exploring the Depths: A Comprehensive Survey of Deep Learning for Underwater Image Processing**

   - Paper discussed in Chapter 3 and annexed in Appendix A.
   - Luiza Ribeiro Marnet, Yury Brodskiy, Stella Grasshof, Erdal Kayacan, and Andrzej Wąsowski. "Exploring the Depths: A Comprehensive Survey of Deep Learning for Underwater Image Processing."
   - Currently under consideration on the *Journal of IEEE Transactions on Artificial Intelligence*.

2. **Mimir-uw: A multipurpose synthetic dataset for underwater navigation and inspection**

   - Paper discussed in Chapter 4 (Sect. 4.1) and annexed in Appendix B.
   - Olaya Álvarez-Tuñón, Hemanth Kanner, Luiza Ribeiro Marnet, Huy Xuan Pham, Jonas le Fevre Sejersen, Yury Brodskiy, and Erdal Kayacan. "Mimir-uw: A multipurpose synthetic dataset for underwater navigation and inspection." In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6141-6148. IEEE, 2023.

3. **SubPipe: A Submarine Pipeline Inspection Dataset for Segmentation and Visual-inertial Localization**

    - Paper discussed in Chapter 4 (Sect. 4.2) and annexed in Appendix C.
    - Olaya Álvarez-Tuñón, Luiza Ribeiro Marnet, Martin Aubard, László Antal, Maria Costa, and Yury Brodskiy. "SubPipe: A Submarine Pipeline Inspection Dataset for Segmentation and Visual-inertial Localization." In OCEANS 2024-Singapore, pp. 1-7. IEEE, 2024.

4. **Bridging the Sim-to-Real GAP for Underwater Image Segmentation**

    - Paper discussed in Chapter 4 (Sect. 4.3) and in Chapter 6 and annexed in Appendix D.
    - Luiza Ribeiro Marnet, Stella Grasshof, Yury Brodskiy, and Andrzej Wąsowski. "Bridging the Sim-to-Real GAP for Underwater Image Segmentation." In OCEANS 2024-Singapore, pp. 1-6. IEEE, 2024.

5. **Uncertainty Driven Active Learning for Image Segmentation in Underwater Inspection**

    - Paper discussed in Chapter 5 and annexed in Appendix E.
    - Luiza Ribeiro Marnet, Yury Brodskiy, Stella Grasshof, and Andrzej Wąsowski. "Uncertainty Driven Active Learning for Image Segmentation in Underwater Inspection." In International Conference on Robotics, Computer Vision and Intelligent Systems, pp. 66-81. Cham: Springer Nature Switzerland, 2024.

Contents

Acronyms

BALD	Bayesian active learning by disagreement. 11
CNN	convolutional neural network. 18, 23, 34
CNNs	convolutional neural networks. 3, 18, 34
DEAL	Difficulty-Aware Active Learning. 49
FCN	fully convolutional network. 34, 35
fps	frames per second. 21, 39, 45
GAN	generative adversarial network. 20
GANs	generative adversarial networks. 18
IoU	intersection over union. 14, 39, 58, 59, 62, 67, 68, 70, 76, 82
IoUs	intersection over unions. 14
LAUV	lightweight autonomous underwater vehicle. 37, 38
mAP	mean average precision. 21
MC-dropout	Monte Carlo dropout. 5–7, 9, 10, 12, 44, 45, 57, 65, 66, 68, 70, 72, 77, 82, 83, 85
MEAL	Manifold Embedding-based Active Learning. 48
meanIoU	mean intersection over union. 14, 15, 38, 45, 47–52, 57–59, 82
POP	Projection onto Orthogonal Prototypes. 13, 64, 65

RPN regional proposal network. 21

S4AL semi supervised semantic segmentation for active learning. 48

SAM segment anything model. 25

SVM support vector machine. 20, 26

SVMs support vector machines. 20

UMAP uniform manifold approximations and projection. 49

Chapter 1

Introduction

For over a decade, deep learning has been revolutionizing the field of computer vision. A notable example of this is the ImageNet competition, for which the most-well known convolutional neural networks (CNNs) for classification were developed. This includes models such as ResNet [2], which surpassed the human accuracy in the ImageNet dataset many years ago [3]. The success of CNNs encouraged the research and application of deep learning models across various domains that uses image analysis, including equipment inspection [4, 5, 6], autonomous driving [7, 8, 9], medical diagnoses [10, 11, 12], underwater debris detection [13, 14], and underwater pipeline inspection [15, 16]

However, it is crucial to recognize that despite this success, deep learning models come with limitations and challenges. Deep learning is a subset of machine learning, and like machine learning models it is data driven [17]. This means that the development and performance of these models highly depend on the availability and quality of the data used for training [18, 19]. Specifically, deep learning models are known for being data-greedy, requiring vast amount of data to learn effectively [20].

This reliance on large datasets poses a problem in fields where data collection is expensive, laborious, or both. In the medical fields, for instance, obtaining data often involves handling strict regulations to preserve patients privacy [21]. For underwater environments, collecting data is time-consuming, costly, and requires extensive logistical planning [22].

Furthermore, after the data is collected, it must be annotated, which is a labor-intensive process that often requires domain experts [20]. It is important to remember that humans also make mistakes when labeling data, as mentioned by Langlotz and co-authors [23] that report clinically important errors in 3% to 6% of images interpreted by radiologists. Errors in the annotation reduce the quality of the dataset, which in turn reduces the quality of the model trained with these data. Annotating the large volume of data needed for supervised training is particularly challenging, as the increase in the workload of the annotator increases the likelihood of human errors [24]. An option for increasing annotation quality is to send the same sample to be annotated by multiple annotators, though this approach leads to higher costs [24].

Another well-known issue with deep learning models is their overconfidence when processing input data out of the training dataset distribution [25]. This means that the model may produce predictions with high probability even when it has little to no knowledge of the input pattern, leading, e.g., to misclassified results with high softmax values for classification problems [26].

It is also important to remember that different image patterns impose varying levels of difficulty to deep learning models. Although deep learning models outperformed humans in the ImageNet competition, ImageNet images have different patterns them other image datasets, such as medical and underwater surveys. Underwater images suffer from unique issues such as low and uneven illumination, scattering, blur, and color distortion [27, 28]. These problems are primarily caused by physical properties of water. Blue light attenuates much slower than the other wavelengths [27], and the presence of floating particles (often referred to as "white snow") intensifies light scattering further reducing image quality [27]. These conditions result in underwater visibility being about 20 meters, and in many cases shorter than 5 meters, depending on the water turbidity [27].

As in any other domain, training deep learning models with underwater images depends on data availability. Generating underwater datasets is specially challenging. Performing underwater surveys for collecting data is expensive, requiring the use of underwater vehicles, equipped with cameras and lights capable of capturing images with minimal quality, and skilled personal to manage the operations. The surveys are also dependent on the weather and water conditions, which limits the opportunities for gathering data. After the data is collected, it needs to be labeled, what is specially difficult due to the low quality nature of these images. Frequently, the data that needs to be annotated is extremely difficult for human eyes, as exemplified in Fig. 1.1.

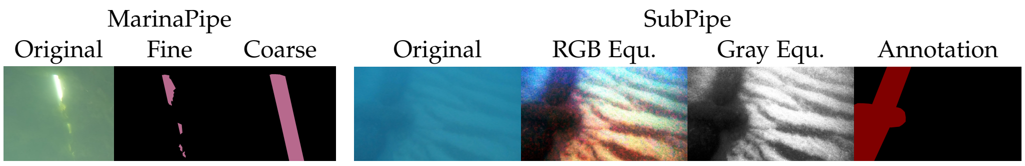


Figure 1.1: The images in this figure are from the MarinaPipe and SubPipe datasets. In these examples, it is possible to observe uneven illumination, color degradation, and blurriness, which reduce visibility. The MarinaPipe dataset received two types of annotation: fine and coarse. Note how the fine annotation is more complex and requires greater precision than the coarse annotation. In the SubPipe dataset, the pipeline images were so difficult to visualize that two preprocessing steps were applied in parallel to assist with labeling. First, histogram equalization was applied to the original RGB images. Second, the original images were converted to grayscale, and histogram equalization was applied. The annotations for the SubPipe images were performed using the visual aid of the dataset equalized in these two ways.

1.1 Motivation and Research Goal

Many researchers [29, 30, 31, 32] studying deep learning models for classification, segmentation or object detection in underwater images highlight the challenges of training models posed by the lack of adequate datasets, as we will further discuss in Chapter 3. They note that existing labeled datasets are often too small and that collecting and annotating sufficient data for their specific tasks is difficult and expensive. Even when data is available, annotating a large dataset is extremely laborious and can be expensive, especially if several annotators are used for decreasing the likelihood of annotation error.

This PhD research primarily focused on testing and developing different strategies to train deep learning models for segmenting underwater images putting in minimal effort on labelling new data. More specifically, the goal was to use epistemic uncertainty calculated with Monte Carlo dropout (MC-dropout) to identify the main gaps of knowledge in the model. This allows for selecting the images for labeling that would better benefit the model's performance improvement when retraining the model. In this research project, we evaluated the use of epistemic uncertainty for distinguishing the predictions of a generalized few-shot segmentation model between trustworthy and unreliable.

Breaking down the main goal into smaller objectives, we worked on:

- Developing datasets for underwater pipeline image segmentation;
- Bridging the gap between models trained with synthetic and real images;

- Active learning using MC-dropout;
- Evaluating the reliability of the predictions of a generalized few-shot learning model using MC-dropout.

1.2 Project Description and Thesis Outline

The first step was to conduct a comprehensive survey on using deep learning techniques for computer vision in the underwater domain, resulting in a survey paper summarized in Chapter 3. This survey covered the use of deep learning models, in the context of underwater monocular RGB images, for image enhancement, classification, segmentation, and object detection. An extensive list of publicly available datasets for these same tasks was also added to the survey.

While listing the datasets, we could observe a lack of publicly available datasets targeted at industrial applications, such as pipeline following or inspection. For this reason, developing this type of dataset would be interesting for advancing underwater image segmentation models for equipment inspection. During this PhD, I contributed to the development and public release of three datasets: MIMIR, SubPipe, and MarinaPipe (Chapter 4).

In the project that generated the synthetic dataset MIMIR, I evaluated the performance of well-known models trained on this dataset for the task of pipeline semantic segmentation (Sect. 4.1). The experiments analyzed the challenges posed by the different environments within MIMIR, the generalization of models trained on one environment and tested in another, and the application of models trained on MIMIR to real-world images. The results showed that, even though models trained on MIMIR could recognize pipelines in a real-world dataset, the performance was limited due to changes in pattern, such as the amount of blur in the real images.

For the project that resulted in the real-world dataset SubPipe, I annotated the images collected during an underwater survey for the task of pipeline segmentation and evaluated the performance of models trained on this dataset (Sect. 4.2). The results highlighted the difficulty for standard models to achieve high performance on the low-quality underwater images, even when segmenting objects with basic shapes, such as straight pipelines. In the MarinaPipe project (Sect. 4.3), I also annotated real images for pipeline segmentation and performed the sim-2-real experiments explained in the the next paragraph and discussed in (Chapter 6).

During a research stay at OceanScan-MST in Portugal, I conducted a research on the gap of knowledge of models trained on synthetic images compared to models trained on real images, Chapter 6. I trained a model using the MIMIR dataset and selected few images from the real-world MarinaPipe dataset for fine-tuning the model and bridge the sim-to-real gap. The results showed that when selecting the same amount of images, the selection based on epistemic uncertainty led to a better set of images for fine-tuning the model than the random selection.

In another research exploring the use of predictive uncertainty, we trained a segmentation model with active learning, Chapter 5. The goal was to train a segmentation model with few image samples that would achieve a result similar to that obtained by the model trained with the entire dataset. The methodology in the project was basically to iteratively select images, based on the uncertainty calculated with MC-dropout, for labeling and retraining the model.

Finally, to address both the data quantity limitation and the deep learning models' overconfidence, we trained a generalized few-shot segmentation model and evaluated the benefits of using epistemic uncertainty for checking the reliability of the model's predictions, Chapter 7.

The use of predictive uncertainty with MC-dropout demonstrates significant potential for reducing annotation time when applied for smartly selecting images. Furthermore, as shown in our experiments in Chapter 7, predictive uncertainty also demonstrates promising results in improving the reliability of the deep learning model predictions. However, these applications occur in offline scenarios. Due to the nature of MC-dropout, which requires several passes of the same input for analyzing the uncertainty, the real-time application requires extending similar approaches as the ones used in this PhD to use predictive uncertainty calculated through other methods, such as the error propagation technique.

Chapter 2

Theoretical Background

This chapter introduces the concepts of predictive uncertainty, active learning, and generalized few-shot learning, which are fundamental for understanding chapters 5 to 7.

2.1 Predictive Uncertainty

Deep learning models are said to be overconfident for assigning high probability to misclassified predictions given input patterns they do not have enough knowledge about. While the softmax values are often used as a measure that reflects the model's confidence for classification tasks, it has been demonstrated that misclassified samples can have high softmax values [26], for example, when an input out of the distribution of the training dataset is used during inference [25]. Therefore, other methods of capturing uncertainty should be used.

The uncertainty of the predictions, called predictive uncertainty, can be decomposed into epistemic and aleatoric [33]. *Epistemic uncertainty* reflects the model's lack of knowledge about the input's pattern, while *aleatoric uncertainty* reflects the data quality itself, e.g. noise caused by the sensor that captures the data. Different methods can be used to estimate predictive uncertainty, e.g. MC-dropout and Deep Ensembles for capturing epistemic uncertainty, direct Modeling for predicting aleatoric uncertainty, and error propagation for both types [34].

MC-dropout is a widely used method for modeling epistemic uncertainty in deep neural networks. It consists of training a model with dropout layers and allowing those layers during inference. Dropout layers, when enabled, set random neurons to zero. During inference, with the dropout layers enabled, the same input sample is forward passed multiple times through the model. Since dropout is applied, each pass may produce a different output, and these outputs are used to assess the epistemic uncertainty. If the model is well-trained and the input is similar to what the model has seen during training, the outputs will be similar or identical. On the other hand, if the model does not have enough knowledge about an input, the outputs of each forward pass will be more varied, indicating high uncertainty. Different metrics, such as variation ratio, mutual information, total variance, margin of confidence, and predictive entropy, can be used for calculating the uncertainty using the MC-dropout outputs [35].

With ensembles of networks, an input sample is passed through the model and the results of each network in the ensemble are used to estimate the uncertainty [36]. In direct modeling, the uncertainty is modeled as one of the model's outputs. For predicting the aleatoric uncertainty, the model needs to be modified to have one output head to predict the uncertainty, and the loss needs to be designed to train the model to learn that. Finally, the error propagation method [37] estimates the model uncertainty by propagating the variance of each layer to the output. This variance arises in layers such as dropout and batch normalization and is modified by the other layers of the model.

In this thesis, we are mainly concerned with capturing the lack of knowledge of the models. For this reason, we use MC-dropout for evaluating the epistemic uncertainty. We use mutual information in chapters 5 and 6 and both mutual information and entropy in Chapter 7.

2.1.1 Entropy and Mutual Information with Monte Carlo Dropout

As mentioned before, entropy and mutual information are two metrics that can be used to calculate epistemic uncertainty with MC-dropout.

In a classification task where each class is represented by an output neuron, whose values sum is equal to 1, the entropy $\mathcal{H}$ is calculated as:

$$\mathcal{H}(p^*) = \frac{-1}{\log_2(C)} \sum_{c=1}^{C} p_c^* \log_2(p_c^*), \tag{2.1}$$

where the prediction for each forward pass t is the softmax output $p_t = (p_{1t}, ..., p_{Ct})$ in a classification problem with C classes, and $p^* = (p_1^*, ..., p_C^*)$ is the average prediction over the T forward passes:

$$p_c^* = \frac{1}{T}\sum_{t=1}^{T} p_{ct}, \tag{2.2}$$

where p_{ct} is each scalar value (output of each neuron c) of the vector p_t in each forward pass t.

The mutual information is calculated as:

$$\mathcal{I} = \mathcal{H}(p^*) + \frac{1}{T\log_2(C)}\sum_{c=1}^{C}\sum_{t=1}^{T} p_{ct}\log_2(p_{ct}). \tag{2.3}$$

We divided these equations by $\log_2(C)$ to normalize the entropy between zero and 1. In this thesis we use $T = 50$ forward passes, following Kendall and Gal [38], and validate this choice in Appendix E.

2.2 Deep Active Learning

Active learning aims to select a small sample set that can be used to train a model to achieve the same performance as when training with the entire dataset [20, 39]. This typically involves training the model with few initial samples and selecting additional samples for labeling and retraining the model iteratively [39]. It is known that random selection usually does not perform well. In active learning with classical machine learning methods, it is common to select new samples using a metric for capturing uncertainty and another for measuring similarity. The former identifies samples for which the model is unsure about the predicted output, and is used for selecting the most informative samples for the model to learn from. The last aids in selecting representative samples and preventing the selection of redundant samples with similar information.

Thus, the first step in developing an active learning framework is to decide on an uncertainty metric. Deep active learning has been applied to tasks requiring costly labeling, such as medical image analysis. The models' prediction values with a single forward pass were used to evaluate the segmentation uncertainty for pulmonary nodules [40] and membrane [41] images, even if these may not be ideal for deep learning models. Other studies have proposed more reliable approaches. Using different subsets of training samples of biomedical images, a set of segmentation models was trained, and the uncertainty was measured as the variance between their outputs [42]. Different metrics, such as max-entropy and Bayesian active learning by disagreement (BALD), were calculated using

MC-dropout outputs for selecting new images to label for training skin cancer image classifier [43] and to segment medical images [44].

More recently, active learning has been studied in the autonomous driving context. The Deeplabv3+ architecture with a Mobilenetv2 backbone was trained with uncertainty- and difficulty-driven image selection [45]. A further reduction of labeled pixels was obtained by querying image patches [46, 47]. Even though these studies used entropy-based metrics for image selection, the metrics were calculated using a single pass softmax output and did not use MC-dropout. In this PhD thesis, the mutual information calculated with MC-dropout was used for accessing the uncertainty.

2.2.1 Acquisition Function

For simple classification problems, where each image receives a single label, the acquisition function can be defined using a metric for calculating uncertainty, such as mutual information or entropy, plus a defined threshold. Images above this threshold should be selected. However, for segmentation, each pixel receives a class label. In this case, each pixel has a prediction that contains a different uncertainty. When querying images, each image needs to receive an aggregated uncertainty value. An option for defining the uncertainty of the entire image, EU_{img}, is to calculate the average uncertainty of all pixels belonging to the image:

$$EU_{\text{img}} = \frac{1}{N} \sum_{i=1}^{N} EU_i, \tag{2.4}$$

where EU_i is the epistemic uncertainty for pixel i of an image with N pixels, calculated with Eq. (2.3).

The threshold TR above which the images should be queried can be defined in different ways. For example, it could be defined as the value that marks the 90th percentile of the calculated uncertainties, querying the 10% images with higher uncertainty. In this PhD thesis, TR was defined as:

$$\text{TR} = \overline{EU}_{\text{img}} + S\sigma, \tag{2.5}$$

where $\overline{EU}_{\text{img}}$ and σ are the mean and standard deviation of EU_{img}, for the corresponding labeled dataset used for training the model, and S is a positive constant defined by the user. Higher values of S result in querying fewer images.

2.3 Few-Shot Segmentation

Well-established deep learning models typically require vast amounts of annotated data for effective training [39, 20]. Moreover, it is usually necessary to define the classes that the model should know in advance. If a new class needs to be recognized after the training is finished, the model must be retrained with the already used dataset (with the base classes) plus a vast amount of samples of the new class. Few-shot learning emerges as a technique to learn a new class using only a few examples [48]. Talking specifically about image segmentation, few-shot segmentation is the field that studies few-shot learning for image segmentation [49].

Prototypical models are one of the most famous neural networks for few-shot segmentation [49]. Prototypical neural networks learn prototypes, in an embedding space. Prototypes are vectors that represent each class. The model uses these prototypes for classification of the images or pixels, depending on how far they are from a given prototype. It is common to train prototypical models with the episodic training paradigm, which simulates the use of the model for new classes during inference. It uses a set of support samples to learn to generate prototypes that represent the classes in a query sample. In each training episode, different classes are used, making the model to learn how to learn representative prototypes instead of learning the classes specifically.

Although few-shot segmentation allows for fast adaptation to new classes with few examples, it does not perform well in recognizing new and previously learned classes simultaneously. Typically, these models only recognize the classes contained in a support sample [49, 50]. This inability to recognize both new and old classes simultaneously is problematic in industrial applications, where it is desirable to learn new classes without forgetting old ones. Incremental [51] and generalized [50] few-shot learning methods aim to address this issue by learning new classes, using only a few examples of the new classes, while maintaining the knowledge of previously known classes. Basically, incremental few-shot learning passes thought several learning sessions, learning new classes in each of them, while generalized few-shot learning could be seen as a special case of incremental learning [51], where the model is only submitted to learn the base classes plus one single session of learning new classes.

Liu *et al.* proposed Projection onto Orthogonal Prototypes (POP) to solve the forgetting problem in a generalized few-shot segmentation task by inserting constrains in the structure and adding a fine-tuning phase to adapt to new classes. For helping with the forgetting issue, they enforced the prototypes to be orthogonal to each other, and trained the model with enough samples of base classes. Later, they added new prototypes for new classes, and froze the feature extractor before fine-tuning for the new classes using only few shots. Later in this thesis, we will use an adapted version of this model for analyzing predictive uncertainty in generalized few-shot segmentation models.

2.4 Metrics for Evaluating Segmentation

It is essential to choose appropriate evaluation metrics for assessing and comparing models. Pixel-wise accuracy is a metric used for evaluating segmentation tasks, and is calculated as the number of pixels correctly classified by the model divided by the total number of pixels [52]. However, it is not a good metric for evaluating the performance of models on datasets with class imbalance, which is often the case. For dealing with this problem, an improved metric is the mean pixel accuracy [52], which can be calculated as:

$$meanAcc = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i} \quad , \tag{2.6}$$

where C is the number of classes, TP_i (true positive) is equal the number of pixels correctly classified as belonging to class i, FN_i (false negative) is the number of pixels belonging to class i and classified as another class, and $TP_i + FN_i$ is equal the total number of pixels belonging to class i according to the ground truth.

Another metric for evaluating segmentation models is the mean intersection over union (meanIoU). The intersection over union (IoU) is calculated for each class as the number of pixels overlapped between the prediction and the ground truth (if both images were placed one on top of the other) divided by the number of pixels in the union of both. The meanIoU is calculated as the mean of the previous calculated intersection over unions (IoUs) per class [52]. The equation for the meanIoU can be written as:

$$meanIoU = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i + FP_i} \quad , \tag{2.7}$$

where FP_i (false positive) is the number of pixels predicted as class i by the model, but that belongs to another class. The meanIoU is the most largely used metric for evaluating segmentation models. Since it also accounts for false positives when evaluating the model, it is a more comprehensive metric than the mean pixel accuracy.

Chapter 3

State-of-the-Art in Computer Vision for Underwater Images

Underwater images suffer from problems not faced by in-air images. Light energy absorption and scattering, caused by the water, limit the visibility of these images [27, 28]. Scattering causes blurring and low contrast, while floating particles in the water (frequently referenced as ’white snow’) increase these problems [27]. Additionally, different light wavelengths attenuate at varying rates. Blue light, for example, travels farther before attenuating as much as other colors [27, 53]. This last phenomenon is responsible for the bluish images. All these factors raise the question of how well deep learning models perform on underwater images.

Before diving into experiments in this PhD project, we developed a comprehensive survey paper (see Appendix A) on deep learning for underwater monocular RGB image processing. This survey provides a thorough overview of recent developments in the field, covering deep learning models used for image enhancement, classification, segmentation, and object detection. We also included a list of publicly available underwater image datasets for training deep learning models.

This survey helped us understand what other researchers have developed and identify a gap in the literature we aimed to further explore. The sections below summarize the key aspects observed during the development of the survey paper.

3.1 Image Enhancement

Enhancing an images means improving its quality in a way that makes features and objects more visible, e.g., increasing the contrast, removing blur or correcting the color degradation. Various approaches have been applied to enhance images using deep learning methods. Simple CNNs trained with supervised learning on matching pairs of original and enhanced images (see Fig. 3.4 for examples of paired images) have been used to improve contrast and illumination, reduce color distortion [54], and remove haze [55, 56]. In a slightly different approach, pairs of real underwater images and color-enhanced images were used to learn the parameters of an underwater image formation equation as part of a model used to enhance images [57].

Generating these datasets with matching pairs of high-quality and low-quality images for training supervised models is challenging. It often requires to synthetically enhance the images to create the dataset, which do not necessarily results in images with the same pattern as naturally good-quality images. To address this limitation, some authors trained generative adversarial networks (GANs) with unpaired datasets, aiming to distort images [58], to transfer style from underwater to in-air [59, 60], and to enhance images with the human perception of image good quality [61].

The Retinex theory [62], which postulates that an image can be decomposed into reflectance and illumination maps [63], has also been employed with CNNs. Han and coauthors [64] trained a convolutional neural network (CNN) to learn the relationship between images and their illumination maps, using these maps to enhance the images.

Zhang et al. [65] analyzed the correlation between image enhancement and object detection performance. These authors argue that the small improvement in object detection does not justify the effort of using image enhancement. Following the idea that visually enhancing the images may not result in the best images for detection models, some authors developed models that simultaneously learn to enhance images and detect objects, ensuring that the enhancement benefits the object detection task [66, 67, 68].

Figure 3.1 shows some images enhanced by the referenced methods. Observe, for instance, that although the image enhanced with the method by Han *et al.* [64] is brighter than the original, the light gets too intense in some regions, making objects less visible. Conversely,

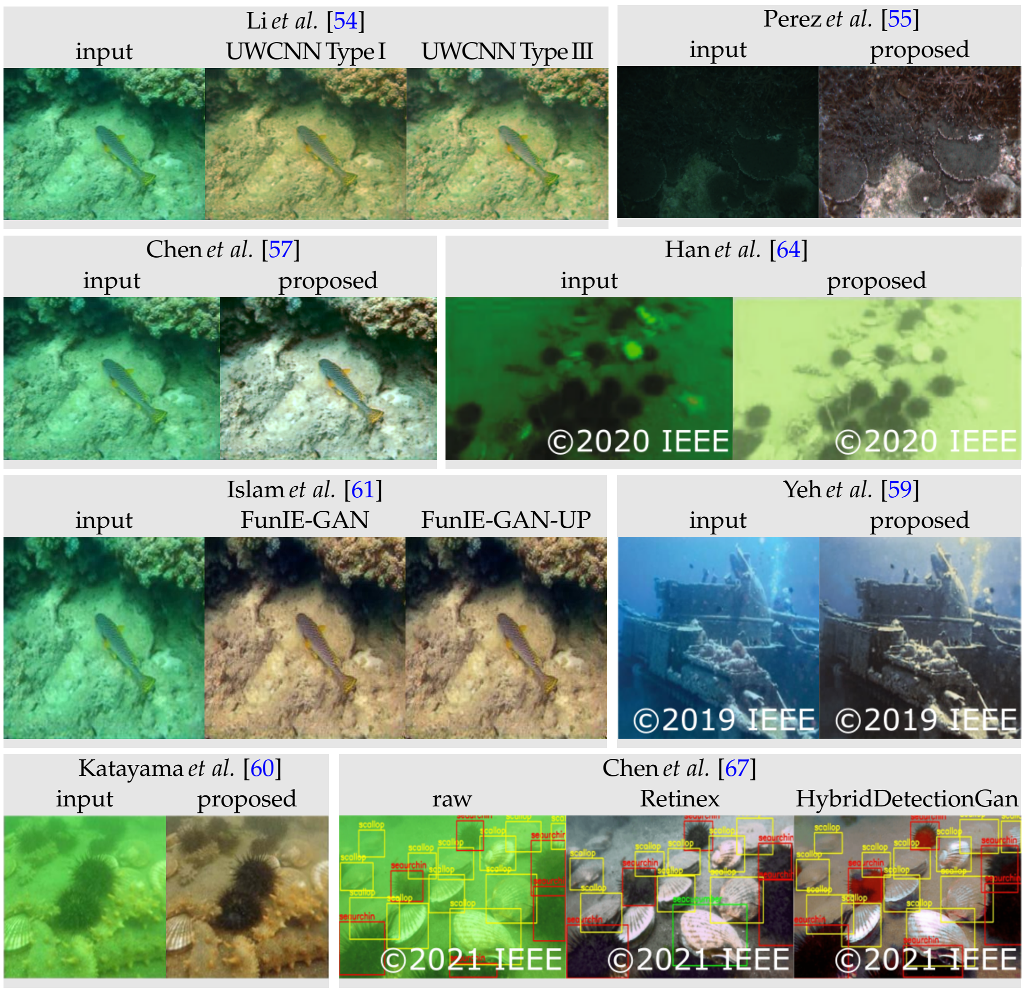


Figure 3.1: Underwater image enhancement. 'Input' are the original images, 'raw' is the input with object detection ground truth. The images were reproduced by us with publicly available code or models, or found directly in publicly available datasets. The work of Han *et al.* [64] is licensed under CC-BY-4.0.

the object detection task is more accurate in the image enhanced with HybridDetectionGan [67], a detection favorable enhancement method, than with Retinex.

3.2 Classification

Image classification is the task of assigning a discrete label to each image. Training deep learning models from scratch typically requires a large amount of data, which is difficult to acquire. To address this problem, researchers from various fields have explored transferring the knowledge from pre-trained models available on the internet, such as models trained on ImageNet [69], and fine-tuning these models with their images.

Cao *et. al* [70] combined hand-crafted features with features extracted from a pre-trained AlexNet [71] to train support vector machines (SVMs) [72] for classifying a fish and a benthic animals dataset. The authors claim that the hand-crafted features helped with the model's robustness when applied to lower-quality images. Similarly, Mahmood *et. al* [73] used a combinations of features extracted from a pre-trained ResNet [2] to train a support vector machine (SVM) [72] to classify bethic datasets.

Moving in the direction of a pure deep learning approach, Jin and Liang [30] pre-trained a model similar to AlexNet [71] on ImageNet and fine-tuned it to classify fish species. Szymak *et al.* [74] used transfer learning to classify images into divers, unmanned underwater vehicles, fish, and water. Between the various pre-trained models and training algorithms tested by Szymak *et al.*, the best combination was Adam algorithm [75] with DenseNet-201 [76].

Using a different method for dealing with small underwater datasets, Chen *et al.* [29] applied adversarial learning for training with a vast amount of in-air labeled images and only few underwater annotated images. Xu *et al.* [77] used a generative adversarial network (GAN) to triple the number of underwater images in the dataset, increasing the accuracy of GoogLeNet [78] trained with this data.

This compilation of research indicates that deep learning performs well on underwater images. However, how to deal with small datasets seems to be a common concern.

3.3 Detection

Detecting objects differs from image classification in that it involves not only assigning a class label to an object but also determining the object location within the image. Unlike classification, which assigns classes to the entire image, object detection assigns classes to the specific regions corresponding to the detected objects. It is common to detect multiple objects, often from different classes, in the same image. Similar to underwater image classification, for making it feasible to train models with relatively small datasets, in object detection it is common to use models pre-trained in large datasets and fine-tune them using underwater images.

Wang and Samani [79] compared transfer learning with training from scratch for the use case of YOLOv3 [80] trained to detect two species of fish. They initially trained the model with 200 images, then trained with 200 more, and finally tested to train YOLOv3 pre-trained with ImageNet. The mean average precision (mAP) increased in each stage. Some researchers have drawn inspiration from how humans learn, first learning easier concepts and then progressing to more complex knowledge. Yuan and co-authors [81] initialized a faster R-CNN [82] with imageNet weights and trained it to detect fish in images with high resolution, contrast and sharpness. Then, the deeper layers of the model were retrained using images with smaller resolution, less contrast, and more blur. Similarly, Chen *et al.* [83] developed a training paradigm called Curriculum Multi-Class Adaboost (CMA), which trains a network discouraging the learning of objects that the model fails to detect, based on the hypothesis that these objects are probably noisy samples and harm the model's learning. The trained model is then used to initialize the training of several detectors. In this last training phase, undetected objects are given more weight. The final detectors are combined into an ensemble.

Several authors have explored applying minor modifications to well-known pre-trained models. Jiang and Wang [84] slightly modified an SSD [85] model by adding two more skip connections between the backbone layers and the detector, aiming to increase the ability to recognize small objects. Ayob *et al.* [86] removed blocks from a MobileNetV2 used as the backbone of a YOLOv2 [87], increasing the detection speed from approximately 12 to 56 frames per second (fps), while reducing the mAP in only 6%. Other researchers have tested mixing features. Fan *et al.* [88] mixes features extracted from different layers of ResNet-50 [2] and VGG-16 [89], aiming to generate a backbone that deals better with blur and distortion. Lin *et al.* [90] inserted a RoIMix module between the regional proposal network (RPN) and the Classifier of a Faster R-CNN [82]. The RoIMix module takes two random RoIs from multiple images and combines them, keeping the label of the RoI that contributes more to the mix. This approach could be interpreted as a form of data augmentation.

A next step related to transfer learning, and that is essential for making models suitable for practical applications, is the model's ability to generalize. For testing generalization ability, Xu and Matzner [91] trained a YOLOv3 [80] model to detect fish using two real-world datasets and tested it in a third dataset. As a result, the model failed on detecting fish in the test dataset. These results highlights the challenge in developing models able to generalize and the importance of using a training dataset with the same pattern as the images that the model is supposed to process in the real application. Liu *et al.* [92] attempted to train domain invariant models by developing DG-YOLO, a modified YOLOv3 [80] that detects the objects in the image and classifies the image domain. They used a gradient reversal layer (GRL) to reverse the gradient during backpropagation, training with adversarial learning to teach the model to ignore the domain. Although not directly, Chen and co-authors [93] also utilized adversarial learning. The authors selected the images in which most objects belong to the minority classes, and employed a CycleGan to apply style transfer to generate these images in various color distortions, improving the class imbalance.

A major challenge in object detection is the loss of detailed information, which impacts the detection of small objects and precise localization. To solve this issue, several authors have implemented strategies such as layers resolution increase (Fig. 3.2), combination of feature maps from different layers' levels, and skip connections from shallower to deeper layers. Liu and co-authors [94] developed AquaNet, which uses a depth-wise separable convolution [95] inspired block with varying kernel sizes, bilinear upsampling for increasing the resolution, and addition of shallower layers and upsampled maps. Pan *et al.* [96] developed M-ResNet, which uses upsampling to combine feature maps from different levels in three different combinations for detecting objects in three different scales: small, medium, and large. Chen *et al.* [97] used skip connections and deconvolution to keep track of detailed information and to generate several inputs for the detector module of their SWIPENET model. Zhang *et al.* [98] performed feature mixing between maps from neighboring layers before the detection block.

Another solution that can help detect small objects is employing deformable convolutions, Fig. 3.3. Different from standard convolution layers, the kernels in the deformable convolutions adapt their receptive field according to the input, which can be an excellent strategy for dealing with objects of different sizes. Zhang *et al.* [99] built a ResNet-101 using deformable convolutions and deformable position-sensitive ROI pooling to improve underwater object detection. Fan *et al.* [88] also utilized deformable convolution in their network.

Another technique applied by many researchers working with underwater images is to include attention blocks in their models. Zhang *et*

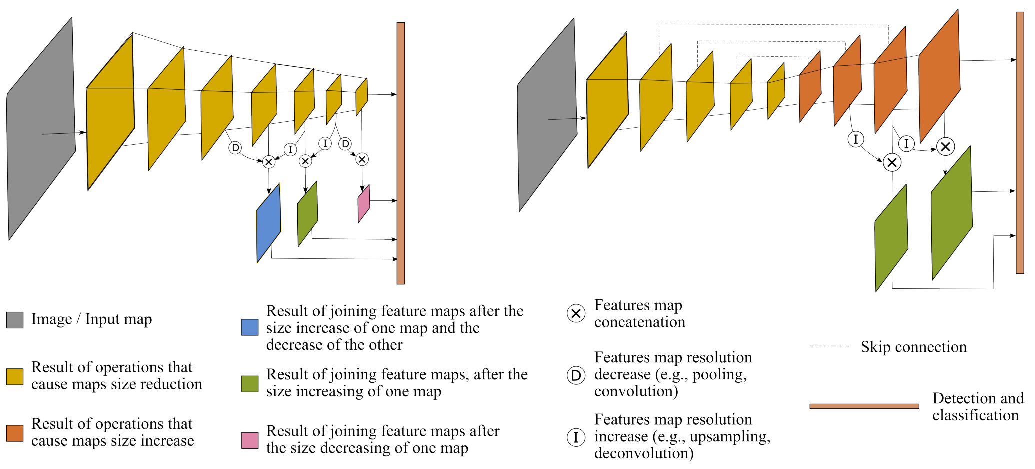

D
I
Image / Input map
Result of operations that cause maps size reduction
Result of operations that cause maps size increase
Result of joining feature maps after the size increase of one map and the decrease of the other
Result of joining feature maps, after the size increasing of one map
Result of joining feature maps after the size decreasing of one map
Features map concatenation
Features map resolution decrease (e.g., pooling, convolution)
Features map resolution increase (e.g., upsampling, deconvolution)
Skip connection
Detection and classification

Figure 3.2: Feature maps combination and increase resolution. The neural network on the left shows different ways of combining feature maps, increasing or decreasing the maps' resolution when necessary. The network on the right shows a resolution increase of the feature maps in 'the main flow' of the CNN. Skip connections are also shown in these images. The models in this figure are a representation to explain the concepts, and do not represent a specific model.

al. [100] developed MFFSSD, a modified SSD [85], which applies spatial and channel attention modules to the outputs of different convolutional layers before combining them and sending the resulting maps to the detector module. Chen and Fan [101] used a mixed attention block after the backbone and trained the resulting detection model with a dataset containing holothurian, echinus, scallop, and starfish. Gao and co-authors [102] added a mixed attention module to the feature extractor of YOLOv4-tiny, improving the ability to detect jellyfish. While these works used typical space and channel attention modules, Ji *et al.* [103] employed a coordinate attention module. The coordinate attention [104] module consists of applying to an input feature map of size (C, H, W) two global average poolings, one with a (H, 1) size kernel and the other with a (1, W) kernel, capturing long-range dependencies along one spatial direction and position information along the other spatial direction.

The works referenced here show concern for improving the detection of objects of different sizes, especially the detection of small objects, and for learning from small datasets, using transfer learning and adversarial learning. Another crucial concern is regarding the quality of data annotation. Given that annotating can be a very laborious and monotonous activity, it is common for annotators to make mistakes or not annotate all the objects in an image. Lv *et al.* [105] developed a training method for

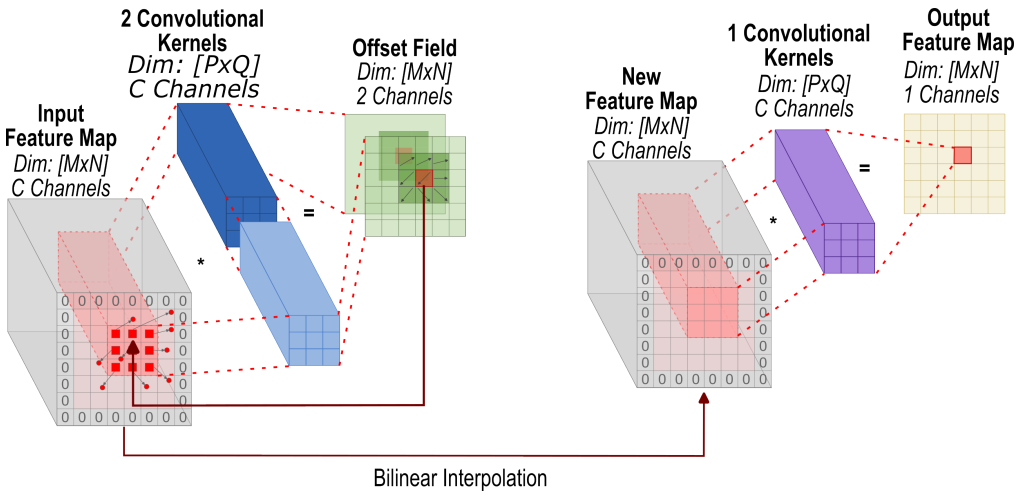


Figure 3.3: Deformable convolution. It learns two 2D offset channels, modifying the standard receptive field in the vertical and horizontal dimensions, adapting the original input feature maps using bilinear interpolation to generate the' new feature map'.

incompletely labeled datasets. Their training strategy removes possibly false negative regions of interest, preventing harming the model's learning. They utilize their method with Faster R-CNN but claim that the same training strategy could be applied to a variety of 2-stage models.

3.4 Segmentation

Segmenting an image is the task of assigning a class label to each pixel. When the segmentation only assigns categorical classes to the pixels without distinguishing between individual objects, it is called semantic segmentation. when it differentiates between classes, it is called instance segmentation. For example, if an image contains three cars, semantic segmentation will simply classify the pixels belonging to all cars as class "car". Instance segmentation, however, will be more specific and classify the pixels as "car 1", "car 2", and "car 3".

In the past few years, numerous segmentation models have been developed, and high performance has been achieved in standard datasets such as PASCAL VOC [106]. Therefore, several researchers have tested the performance of these models in the underwater environment. Thampi *et al.* [107] utilized U-Net [108] to segment images between background and fish. They observed that the best sigmoid thresholds for identifying fish from the five species contained in their dataset were all between 0.5

and 0.6. Drews-Jr *et al.* [109] yielded a research using DeepLabV3+ [110] and SegNet [111]. They observed that DeepLabV3+ achieved better results when initialized with weights pre-trained in PASCAL VOC [106], while SegNet presented slightly better results with no pre-training. The authors associated this observation with DeepLabeV3+ being larger than SegNet. These results emphasize the importance of training large models with vast amounts of data. The authors also tried to pre-process the images using the Underwater Dark Channel Prior method (UDCP) [112] and UGAN [58], but did not achieve encouraging results, which aligns with the earlier observation regarding the correlation between image enhancement and object detection performance.

Continuing with well-known models, Xu and co-authors [113] fine-tuned the segment anything model (SAM) [114] using the underwater dataset SUIM [115], obtaining AquaSAM. Hong *et al.* [116] trained Mask R-CNN [117] to perform instance segmentation of marine trash. Other researchers modified well-established models aiming to improve performance or reduce inference time. Zhao and co-authors [118] modified Mask R-CNN for pipelines and oil leakage segmentation by implementing a multi-layer feature map, which allows better information flow between low- and high-level feature maps, and by adding the boundary-weighted loss function, which increases the ability to segment edges. Islam *et al.* [115] aimed to develop a faster segmentation model. For that, they used an encoder-decoder structure and skip connections between both. In the encoder, they added skip connections inside residual inspired blocks. He *et al.* [119] used a ResNet backbone with an auxiliary network that uses GhostConv [120] for extracting better features in the encoder. In the decoder, they used multi-scale feature fusion with channel attention for not losing information of deeper or shallower feature maps. Additionally, they employed a combination of cross-entropy and dice loss to improve boundary segmentation. Wang *et al.* [121] aimed to achieve better contextual information, and added a pyramid pooling to the output of SegNet. To address the issues of color degradation and low definition in underwater images, they pre-processed the images with rule-based techniques, aiming to stretch the color channels and extract edges with the Sobel filter.

Finally, similar to datasets for classification and object detection, datasets for training segmentation models are relatively small. Consequently, many authors have focused on developing methods for dealing with small datasets. Fan *et al.*[31] trained an adapted U-net [108] using an adversarial transfer learning approach to train the model with ground crack and underwater crack images to learn to predict underwater dam cracks. Saleh *et al.* [122] performed unsupervised training to teach a w-net [123] model to segment images from fish4knowledge. W-net consists of two u-nets [108], where the first U-net is supposed to segment the images, while the second one should reconstruct them. O'Byrne and co-authors [32] trained a SegNet [111] model with photorealistic images of structures containing biofouling, and obtained good results when applying this model to real-world images. The segmentation obtained with SegNet was subsequently improved with an algorithm based on SVM.

Several researchers have utilized clusterization for training segmentation models starting from sparse annotation [124, 125, 126]. The base idea is to start with sparsely annotated images containing only a few annotation points and generate dense masks by clusterization. These dense masks are then used to train deep learning models such as DeepLabV3 and SegNet. This strategy saves annotation time and deals with weakly annotated datasets.

It is evident that deep learning segmentation models perform well. However, researchers working with underwater images continue to face challenges in obtaining sufficiently large annotated datasets, which is necessary for supervised training. To address this issue, several techniques are employed. It is interesting to observe the use of different methods such as adversarial learning, sparse annotation, and photorealistic images. Similar to classification and object detection, using pre-trained models is common for enabling the use of smaller datasets. Furthermore, the use of other techniques also observed in object detection, such as skip connection and attention blocks, highlights how advancements in one field can benefit the other.

3.5 Datasets

While surveying various papers on underwater image processing, we identified several datasets used for training deep learning models for image enhancement, classification, segmentation, and object detection. Table 3.1 lists the publicly available datasets we encountered and the links to where to find them. Figure 3.4 provides some examples of the referenced datasets.

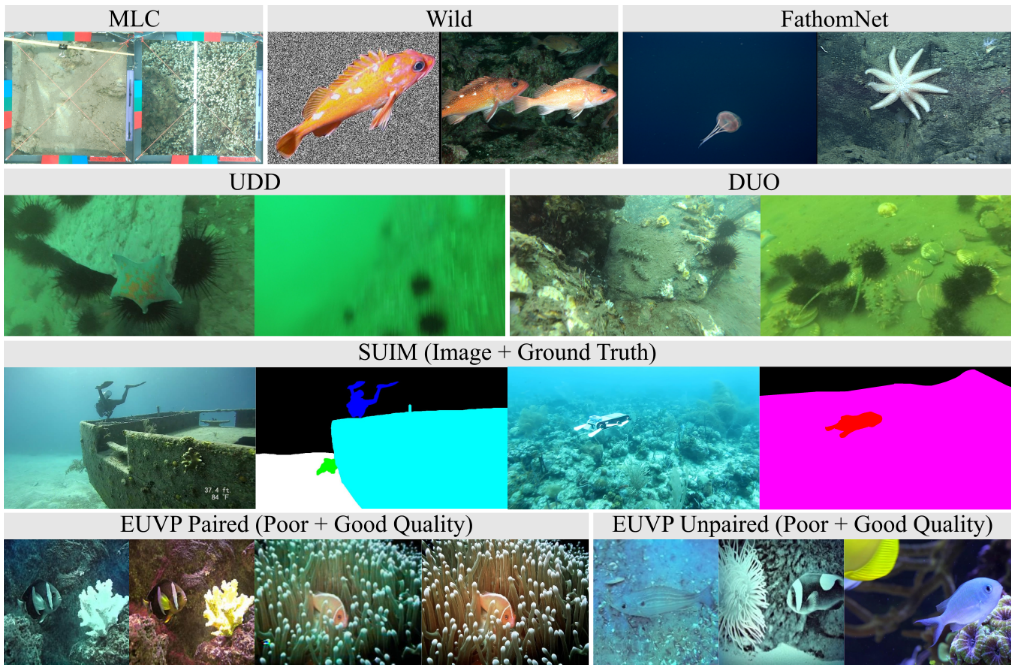

MLC
Wild
FathomNet
UDD
DUO
SUIM (Image + Ground Truth)
EUVP Paired (Poor + Good Quality)
EUVP Unpaired (Poor + Good Quality)

Figure 3.4: Examples from the identified datasets. For SUIM (segmentation) we show the ground truth mask as well. For EUVP paired (enhancement) we include both the original good-quality and distorted (by a CycleGan [140]-based model) images. For EUVP unpaired (enhancement), the first two images are low-quality, and the last one is a better-quality example. MLC dataset is under the CC-BY-4.0 license.

3.6 Discussion

After reading all the surveyed papers, we observed some of the main points explored by researchers, as well as gaps in the literature that we would like to address. Notably, there is a growing interest in developing deep learning models for underwater images, as Fig. 3.5 indicates.

The first essential point to discuss is the difficulty of comparing papers on deep learning for underwater image processing, which is a consequence of various reasons. One major problem, that occurs in some papers, is the need for more details about the models' structure and the training method. Another observed problem is that multiple changes are often applied to the models, many times without individual analysis of how each modification improves the performance. Finally, a recurrent problem that prevents fair comparison between papers is the choice of datasets employed by different researchers. General classification models are usually benchmarked against ImageNet [69], few-shot segmentation models against PASCAL-5i [141] and COCO-20i, and semantic segmentation in autonomous driving and urban scene understanding against

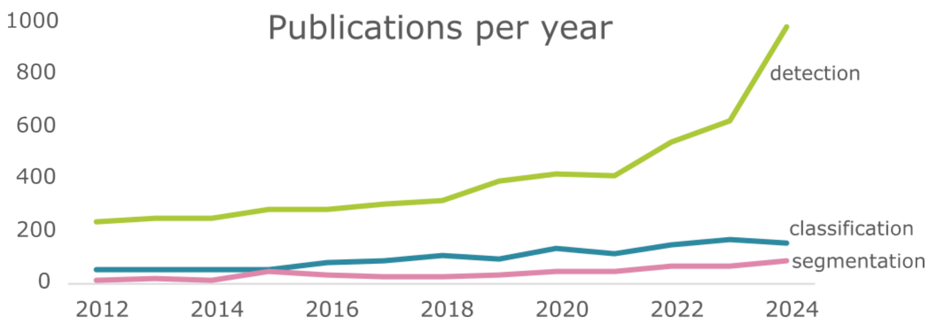


Figure 3.5: Publications rate, according to Google Scholar, for works that include 'underwater,' 'coral,' or 'fish' combined with 'classification', 'detection,' or 'segmentation' in the title. We excluded entries including 'sonar,' 'ultrasound,' 'ultrasonic,' 'acoustic,' and 'acoustics' in the title. For 2024, the number is extrapolated from the amount of publications in the first 5 months of the year.

CamVid [142, 143] and Cityscapes [144]. However, a generally accepted standard dataset does not exist to benchmark deep learning models on underwater images. To accelerate the progress in the field, it would be important that the research community agree on datasets for models comparison. For example, DeepFish could be used for classification, UDD for detection, SUIM for semantic segmentation, and TrashCan for instance segmentation. If researchers have specific datasets of interest, these would be used as an additional test of the model's performance.

An important future research direction is the study of the deep learning models' overconfidence. For solving this overconfidence problem, the predictive uncertainty [145] of the models has been studied in critical fields, such as medical images [146, 147, 148, 149]. However, for the underwater environment, the application of uncertainty in image analysis is new, appearing in the recent research about regularization to count fish in sonar images [150] and to perform classification of heterogeneous underwater soundscapes [151]. Using uncertainty to enhance the reliability of models in underwater environments is a research area that has not yet gained significant attention and should be further investigated.

Finally, the available annotated image datasets for underwater applications are usually relatively small. For examples, the CIFAR-100 [152] dataset contains 100 classes and 60,000 images for classification; COCO [153] includes 80 classes and oven 200,000 annotated images; and PASCAL VOC [106] has 20 classes and nearly 7,000 images annotated for semantic segmentation. In comparison, as shown in Tbl. 3.1, the underwater datasets are usually smaller in number of images and lack diversity , e.g. datasets with 4 classes only, and the shortage of datasets with classes relevant to industrial applications. Generating these datasets is laborious and potentially expensive. To address this issue, several researchers used transfer and adversarial learning. In this PhD, we propose leveraging predictive uncertainty to improve the data annotation efficiency. We explored the gap in the literature by using the predictive uncertainty of deep learning models for computer vision in the underwater domain while trying to tackle the lack of large annotated datasets. Additionally, given the focus on applying this research in the marine industry, and considering the lack of datasets targeting industrial applications, we have also worked on developing datasets for underwater pipeline semantic segmentation.

Table 3.1: Datasets for image enhancement, classification, object detection, and segmentation tasks. Size is given in the number of images

Name (linked)	Task	Size	Contents
EUVP-Unpaired [61]	image enhancement	6665	divided into poor and good quality images, 95% training, 5% validation
EUVP-Paired [61]	image enhancement	13405	divided into poor and good quality images, 85% training, 15% validation
MLC [127]	classification	2055	non-coral classes: crustose coralline algae (CCA), turf algae, macroalgae, and sand; and coral genera classes: *acropora*, *pavona*, *montipora*, *pocillopora*, and *porites*
BENTHOZ-2015 [128, 129]	classification	9874	148 substratum and biological classes
labeled fishes in the wild [130]	detection	4996	fish, invertebrates, and the seabed; 929 positive, 3167 negative samples, 210 video frames of test material
OUC-VISION [131]	detection	4400	salient objects (like rocks). Unfortunately we were unable to locate teh dataset online, despite the authors claiming that it was available. The corresponding author is Muwei Jian (jianmuwei@ouc.edu.cn)
Brackish [132]	detection	14674	fish, small fish, crabs, shrimps, jellyfish, and starfish (80% training, 1468 testing, and 1467 validation)
UDD [94]	detection	2227	sea-cucumbers, sea-urchins, and scallops (1827 training, 400 testing)
DUO [133]	detection	7782	holothurian, echinus, scallop, and starfish (6671 training, 1111 testing)
FathomNet [134]	detection	(growing)	mid-water and benthic objects
TrashCan [116]	detection, segmentation	7212	trash (with object name and material), man-made objects intentionally placed in the scene, bio (animal or plant), and unknown
SUIM [115]	segmentation	1635	background (water), human divers, aquatic plants and sea-grass, wrecks or ruins, robots (AUVs, ROVs, instruments), reefs and invertebrates, fish and vertebrates, sea-floor and rocks; includes 110 test images
DUT-USEG [135]	segmentation, detection	6617	sea cucumber, sea urchin, scallop and starfish (1487 are labelled for segmentation)
NAUTEC UWI [109]	segmentation	700	foreground and background
MarinaPipe [136]	segmentation	1723	pipeline and background, the images are divided into 7 folds, corresponding to frames from 7 videos
SubPipe [137]	SLAM, detection, segmentation	647	pipeline and background, includes side-scan-sonar images for object detection
LIACI [138]	segmentation	1893	defects, corrosion, paint peel, marine growth, sea chest gratings, overboard valves, propeller, anodes, bilge keel, and ship hull
DeepFish [139]	classification, detection, segmentation	620	fish (about 50/20/30% split into training, validation, and test)

Chapter 4

New Underwater Datasets

As discussed in Chapter 3, the lack of datasets is a major obstacle to the fast development of deep learning models for underwater environments. To address this limitation, we contribute to the development and public release of three datasets targeting semantic segmentation and tracking of pipelines in underwater images. Section 4.1 describes the contribution to MIMIR, a synthetic underwater camera images dataset, while Sect. 4.2 and Sect. 4.3 present the work developed with subPipe and MarinaPipe, both datasets containing camera images collected in real underwater environments.

4.1 MIMIR

MIMIR, published in the paper from Appendix B, is a multipurpose dataset tailored to pipeline tracking tasks, including visual SLAM, depth estimation, and image segmentation. This dataset, developed in Unreal Engine 4, comprises four synthetic underwater environments [154]. A simulated robot, deployed in the four environments, collected images across 11 predefined tracks. For each track, RGB images were captured from the point of view of three cameras: two forward-facing stereo cameras (cam 0 and cam 1) and a bottom-looking camera (cam 2). AirSim's image API automatically generated the segmentation masks and depth information for each collected image.

The four MIMIR environments are:

- SeaFloor: A scenario of shallow water with good visibility, but containing light reflectance. This environment contains a set of

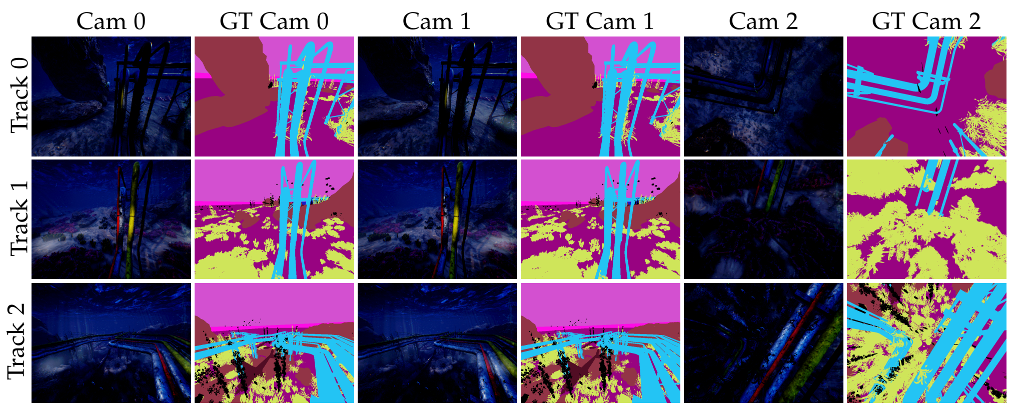


Figure 4.1: Images recorded across the three tracks performed in the SeaFloor environment. For each track, the images provided as example were captured by the three cameras at the same time stamp. GT refers to ground-truth.

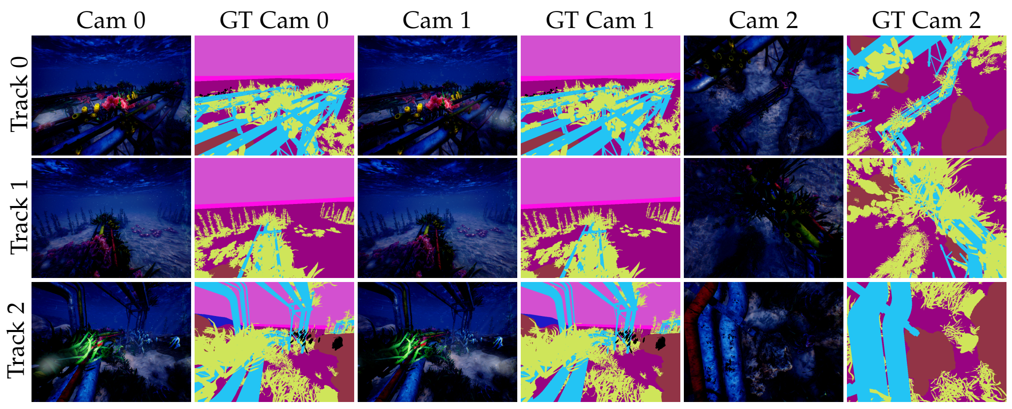


Figure 4.2: Images recorded across the three tracks performed in the SeaFloor Algae environment. For each track, the images provided as example were captured by the three cameras at the same time stamp. GT refers to ground-truth.

relatively thin pipelines (see Fig. 4.1).

- SeaFloor Algae: An extension of SeaFloor, containing a higher number of dynamic objects that partially occlude the pipelines (see Fig. 4.2).

- OceanFloor: A deep-water environment where almost all the visibility in the camera images is due to artificial illumination. This environment also contains a set of thin pipelines. See Fig. 4.3.

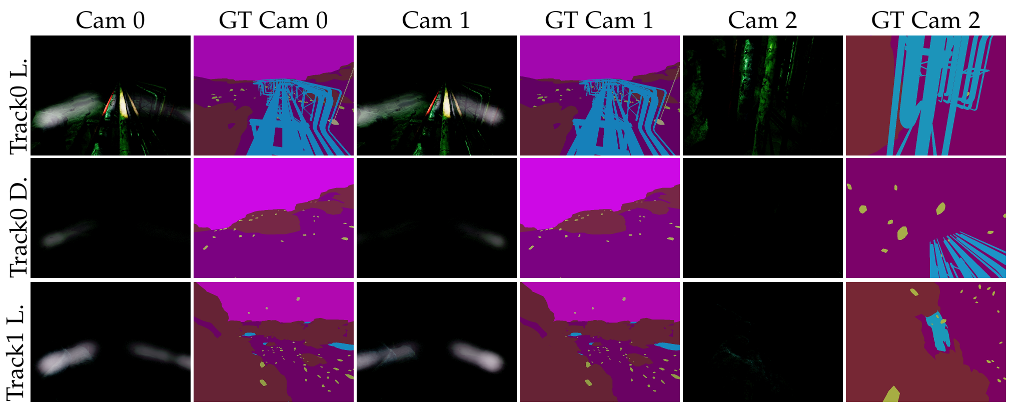


Figure 4.3: Images recorded across the three tracks performed in the OceanFloor environment. For each track, the images provided as example were captured by the three cameras at the same time stamp. GT refers to ground-truth. 'L' and 'D' in 'Track1 L.', 'Track0 L.', and 'Track0 D' refers to light and dark.

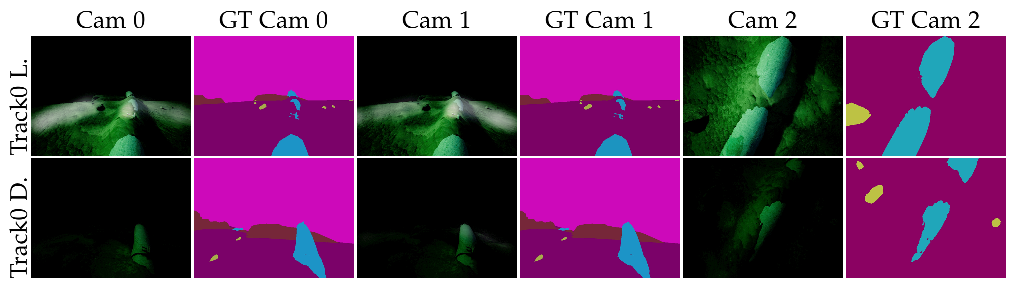


Figure 4.4: Images recorded across the two tracks performed in the SandPipe environment. For each track, the images provided as example were captured by the three cameras at the same time stamp. GT refers to ground-truth. 'L' and 'D' in 'Track0 L.' and 'Track0 D' refers to light and dark.

- SandPipe: Another deep-water environment, but it contains a single larger pipeline, which is mainly covered by sand and visible in parts of the trajectory (see Fig. 4.4).

We trained two segmentation models with images from different environments to evaluate the performance of deep learning models using this dataset. Finally, to assess MIMIR's potential for real-world application, trained models were tested on underwater images collected in real surveys.

4.1.1 Experiments and Results

MIMIR contains segmentation masks for several classes, but we focus our experiments on segmenting pipeline from background. The environments present different levels of difficulty, and we evaluated the performance of the fully convolutional network (FCN) [155] and DeepLabV3 [156], utilizing the PyTorch implementation, across multiple MIMIR tracks. The FCN model consists of a classification CNN as the backbone with the fully connected layers replaced with upsampling and convolutional layers to recover the input image dimensions. DeepLabV3 is a segmentation CNN that uses dilated (also called atrous) convolutions with multiple dilated rates to capture long-range information and improve the segmentation of objects in various scales.

Table 4.1 shows the results of the experiments. The first three only utilized MIMIR images and the others evaluated the generalization of models trained on MIMIR to real data. Experiment 'SeaF.' was conducted using the SeaFloor environment. The first 80% of tracks 0 and 1 were utilized for training, and the last 20% for validating, while track 2 was used for testing. The results for both CNNs were similar. If we observe Fig. 4.1, the three tracks are quite similar and the visibility is not extremely bad, so it was not a surprise that the test performance was relatively good.

The experiment 'SandP.' was performed with the SandPipe environment, with track 'dark' used for training and validation, and track 'light' for testing. The poor results of this experiment are likely explained by the extremely bad visibility of track dark, c.f. Fig. 4.4. The experiment 'All' evaluates the models' generalization across environments, using the

Table 4.1: Experiments setup and results for the segmentation models trained with MIMIR

Exp.	split	**(%) sequence{track}{camera}**	FCN			DeepLabV3		
			bg	pipe	mIoU	bg	pipe	mIoU
SeaF.	Train	80% SeaFloor{0,1}{cam0}	0.941	0.766	0.854	0.941	0.766	0.853
	Val	20% SeaFloor{0,1}{cam0}	0.934	0.685	0.809	0.934	0.686	0.810
	Test	100% SeaFloor{2}{cam0}	0.903	0.667	0.785	**0.905**	**0.674**	**0.790**
SandP.	Train	80% SandPipe{dark}{cam0}	0.936	0.199	0.568	0.953	0.256	0.604
	Val	20% SandPipe{dark}{cam0}	0.832	0.317	0.574	0.823	0.298	0.560
	Test	100% SandPipe{light}{cam0}	0.818	0.073	0.445	**0.898**	**0.146**	**0.522**
All	Train	80% SandPipe,SeaFloor,OceanFloor{all}{cam0}	0.930	0.618	0.774	0.934	0.636	0.785
	Val	20% SandPipe,SeaFloor,OceanFloor{all}{cam0}	0.887	0.460	0.674	0.889	0.458	0.674
	Test	100% SeaFLoor Algae{all}{cam0}	0.888	0.512	0.700	**0.893**	**0.515**	**0.704**
S2R SandP.	Train	80% Sandpipe{light}{cam2}	0.993	0.702	0.847	0.991	0.658	0.824
	Val	20% Sandpipe{light}{cam2}	0.977	0.851	0.914	0.976	0.849	0.913
	Test	Real Pipeline Dataset	0.477	**0.219**	0.348	**0.535**	0.214	**0.374**
S2R Aug. SandP.	Train	80% Sandpipe{light}{cam2}	0.978	0.429	0.703	0.977	0.419	0.698
	Val	20% Sandpipe{light}{cam2}	0.924	0.637	0.780	0.903	0.579	0.741
	Test	Real Pipeline Dataset	0.713	0.502	0.607	**0.739**	**0.531**	**0.635**
S2R Aug. SeaF.	Train	80% SeaFloor{0,1,2}{cam2}	0.909	0.753	0.831	0.898	0.738	0.818
	Val	20% SeaFloor{0,1,2}{cam2}	0.901	0.733	0.817	0.886	0.711	0.798
	Test	Real Pipeline Dataset	**0.703**	0.208	**0.455**	0.554	**0.356**	**0.455**

SeaFloor Algae environment for testing, and all the other environments for training and validation. The results were better than for the second experiment, but still much lower than the results for ′SeaF.′. SeaFloor and SeaFloor Algae are very similar environments, and training and validating with images from three environments increases data variability, which should diminish overfitting. Yet the algae covering the pipes in SeaFloor Algae is a new challenge for the model.

The final three experiments evaluate the generalization of the models to real-world images, which assess MIMIR's potential for practical applications. The real-world dataset utilized in these experiments belongs to EIVA. It was acquired by their partners, and is not publicly available for security reasons. The ′S2R SandP.′ model was trained and validated on the SandPipe Light track without using data augmentation, whereas in ′S2R Aug. SandP.′, we utilized random flips and resize, random changes in brightness, contrast, and saturation, addition of Gaussian blur, and the images were converted to grayscale. These techniques mitigate overfitting to MIMIR and diminish the gap between the synthetic and real images. Table 4.1 shows that these augmentations significantly improved the pipeline detection. Even though the same augmentations were utilized in the ′S2R Aug. SeaF.′ experiment, the results on the test real-world images were much lower than in ′S2R Aug. SandP.′. This result can probably be explained by the fact that SeaFloor is much more different from the real images utilized in these experiments than SandPipe, as can observed in Fig. 4.5.

Figure 4.5 shows three results of the ′S2R Aug. SandP.′. The image in the first example is the sharpest one and contains a single pipe on the bottom floor, such as in the SandPipe environment. Between the shown examples, this was the best image segmented by the models. In comparison, the image in the second example contains much more blur than the MIMIR images, and the detection is much worse. In the third example, the models only detect the largest pipeline in the image, possibly because SandPipe contains a single pipe. In the majority of the results from Tbl. 4.1, we did not observe big differences in performance between DeepLabV3 and FCN. In the Sim2Real experiments, for pipeline detection, the difference in performance is more evident. The combination of dilated convolutions employed by DeepLabV3, which improves the ability to capture spatial information, is possibly responsible for the better performance of this model - which is an important factor

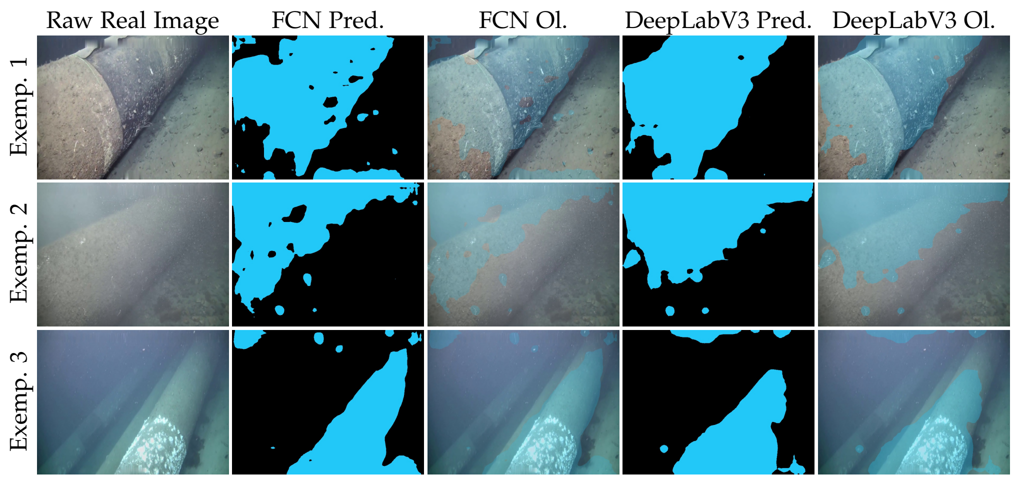


Figure 4.5: Examples of the results obtained with the 'S2R Aug. SandP.' experiment. 'Pred.' are the models' predictions and Ol. are the overlapping of the predicted masks with the raw images.

since pipeline represents about 10% of the pixels in MIMIR and occupies smaller regions of images than the background.

The environments in the MIMIR dataset are complex, and the results of the experiments show that simply training well-known models does not completely solve the semantic segmentation problem. Additionally, the results of the experiments testing the models' generalization capabilities highlight that this is not a trivial task. MIMIR is particularly interesting because it contains data from similar tracks that vary in illumination and amount of dynamic objects, simulating data collected in different underwater surveys at similar locations. This makes this dataset valuable for developing models with better generalization capability targeting real applications. It was also observed that, although MIMIR is highly realistic, it is much sharper than real images. This difference affects the performance of a model trained on MIMIR when applied to real images. However, this dataset can be used for pre-training models that will later be fine-tuned with real images to adjust the gap in patterns, as we will further explore in Chapter 6.

4.2 SubPipe

Subpipe is a dataset of underwater images, developed in collaboration with OceanScan-MST and designed for SLAM, object detection, and image segmentation (see Appendix C). It was recorded using a physical

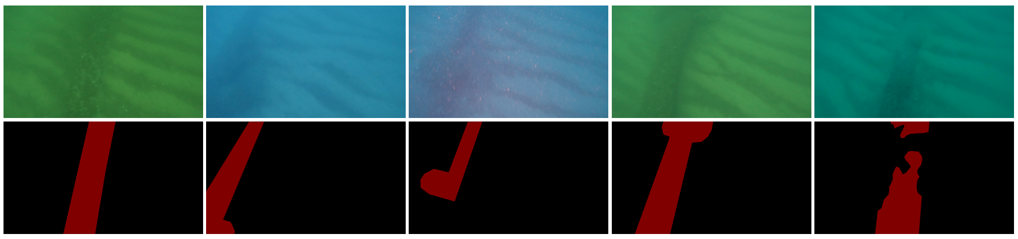


Figure 4.6: SubPipe images. Top: Original raw images; Bottom: Manually annotated Ground-truth

lightweight autonomous underwater vehicle (LAUV) along an approximately 1km long underwater pipeline in northern Portugal, at depths of up to 18 meters. This survey recorded data from several sensors, including grayscale and RGB camera images, side-scan sonars, temperature, pressure, and depth measurements. SubPipe contains data from five recorded videos, and about 20% of this data was used to create a lighter version of the dataset, named SubPipeMini.

SubPipe was developed for pipeline image segmentation. Segmentation masks for the RGB images were manually generated using the LabelMe tool [157]. The images in SubPipe are semantically repetitive, displaying the ocean floor with a pipeline crossing the image, but with various levels of color degradation. Due to the data redundancy, annotation was generated to every 25th frame of SubPipeMini only, resulting in 647 annotated images.

Figure 4.6 presents sample images from SubPipe, illustrating the low visibility caused by the significant color degradation and poor illumination. This low visibility makes manually labelling the images difficult. To improve the annotation quality, reducing errors, two pre-processing steps (Fig. 4.7) were applied in parallel: (1) histogram equalization of RGB channels and (2) grayscale conversion followed by histogram equalization. Labels were generated by observing the three sets of images: original, RGB histogram equalized, and grayscale equalized. The pre-processed images were used to assist in the annotation but were not released with the dataset.

To evaluate how challenging this dataset is for pipeline segmentation task, two models were trained: SegFormer [158] and DeepLabV3 [156]. SegFormer is a segmentation model with transformer blocks in the encoder, which uses the multi-scale outputs from the encoder as input to a multilayer perceptron decoder. SegFormer was trained from scratch using the implementation by Zuppichini [159]. DeepLabV3 was trained

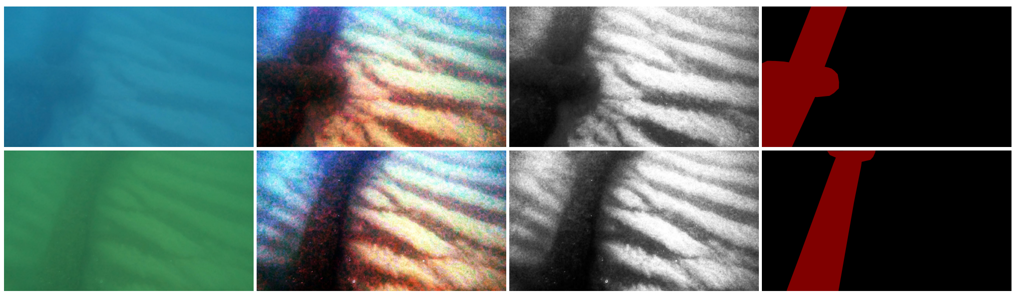


Figure 4.7: SubPipe annotation process. From left to right: (1) raw images; (2) images after histogram equalization; (3) histogram equalization of raw images converted to gray scale; (4) ground truth generated by visually analyzing images from columns (1-3).

with the official Pytorch implementation. The first 60% of the annotated data was used for training, the next 20% for validation, and the last 20% for testing.

The results are presented in Tbl. 4.2. Both models were able to segment the pipeline, even with SegFormer trained from scratch, indicating that the dataset has enough size and quality for training deep learning models. However, despite the pipeline's relatively simple shape, the performance of the best model - SegFormer - achieved only 66.4% meanIoU for the pipeline class. The experiment performed with SubPipe could be compared to the 'SeaF.' experiment in Sect. 4.1, where images from the same environment were used for training, testing, and validation. However, although the environment SeaFlor in MIMIR is much more complex than SubPipe, which contains only a single pipeline on a sand bottom floor, DeepLabV3 performed much better in the 'SeaF.' experiment. These results are likely due to the amount of blur and levels of color degradation in the images, highlighting the challenges posed by images collected in real underwater surveys. SubPipe could be used for developing deep learning models for the underwater environment that are robust to these problems.

4.3 MarinaPipe

MarinaPipe is a dataset of underwater images, recorded in a marina near northern Portugal by our partner, OceanScan-MST (see Appendix D). The dataset consists of images of pipes placed on the marina floor, filmed using a facing-down GoPro camera mounted on the bottom of an LAUV.

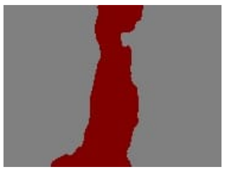
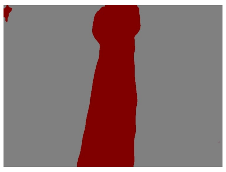
Seven videos were recorded at 240 fps, from which we extracted five frames per second to create the dataset. All videos were recorded on the same day, within short time frame. In some videos, the pipes are partly occluded by algae.

For performing the experiments described in this section and in Chapter 6, 10% of the frames of each video were labeled for the task of pipeline segmentation. Table 4.3 provides an overview of the MarinaPipe dataset. This dataset was originally idealized for training a model that would later be tested for tracking long pipelines. Because of this, the extracted frames were cropped before being labeled, so that the pipe goes through the image, as Fig. 4.8 shows. We released the original videos, the extracted frames (Tbl. 4.3), and the fine and coarse pixel-wise annotation for the pipeline class in the format of masks.

The MarinaPipe images were used for training the same visual transformer, SegFormer [158], used for training SubPipe in Sect. 4.2. For the MarinaPipe experiments, we modified the code written by Zuppichini [159] to include dropout layers in the encoder. The annotated frames from videos 1 and 7 were used for training and validation, respectively. Table 4.4 presents the performance of the trained SegFormer when comparing the predictions both with coarse and fine annotation.

The low IoU for the pipeline class, shown in Tbl. 4.4, indicates that MarinaPipe is a very challenging dataset. As can be observed in figures 4.8 and 4.9, even for human eyes it is difficult to define the precise edges of the pipelines in the images. It is worth noticing that the model was trained and validated with videos 1 and 7, which contain pipeline occlusion, and that it completely failed to segment the pipeline in videos

Table 4.2: Experiments results for the segmentation models trained with SubPipe

Image	Ground Truth	SegFormer	DeepLabV3
			
Results	Background	**0.918**	0.912
	Pipeline [%]	**0.664**	0.601
	mIoU [%]	**0.791**	0.757

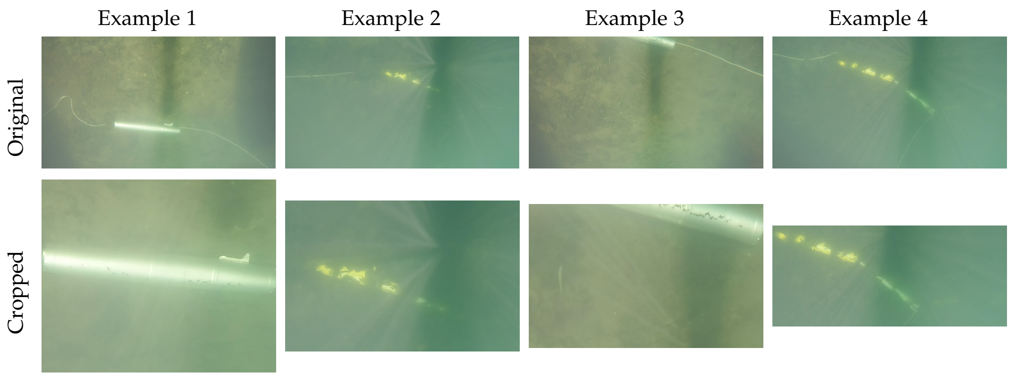


Figure 4.8: Examples of how the video frames were cropped before being labeled. Top: Original frames. Bottom: Correspondent cropped frames.

2 and 3, which do not contain occlusions. This observation highlights the importance of training the model with data that matches the patterns in the images to which the model will be later applied.

As a final analysis, observe that although the model was trained with the fine annotation as ground truth, the performance was higher when evaluated against coarse annotation. This outcome suggests that the

Table 4.3: Details of MarinaPipe. (*Both* refers to fine and coarse labeling.)

Video	Selected frames	Frames with pipes	Annotation	Occlusions
1	236	43	Both	Yes
2	237	70	Both	No
3	260	2	Both	No
4	268	11	Both	Yes
5	266	45	Coarse	Yes
6	270	11	Coarse	Yes
7	186	17	Both	Yes

Table 4.4: Performance obtained by SegFormer trained with MarinaPipe. Videos 1 and 7 were used for training and validation, and all the others for testing only.

Video	Fine Annotation			Coarse Annotation		
	Pipe	Background	meanIoU	Pipe	Background	meanIoU
1 (training)	0.093	0.989	0.541	0.335	0.989	0.662
2	0.000	0.921	0.460	0.000	0.921	0.460
3	0.000	0.994	0.497	0.000	0.994	0.497
4	0.038	0.994	0.516	0.124	0.99	0.559
5	-	-	-	0.231	0.984	0.608
6	-	-	-	0.026	0.962	0.494
7 (validation)	0.018	0.971	0.495	0.090	0.970	0.530

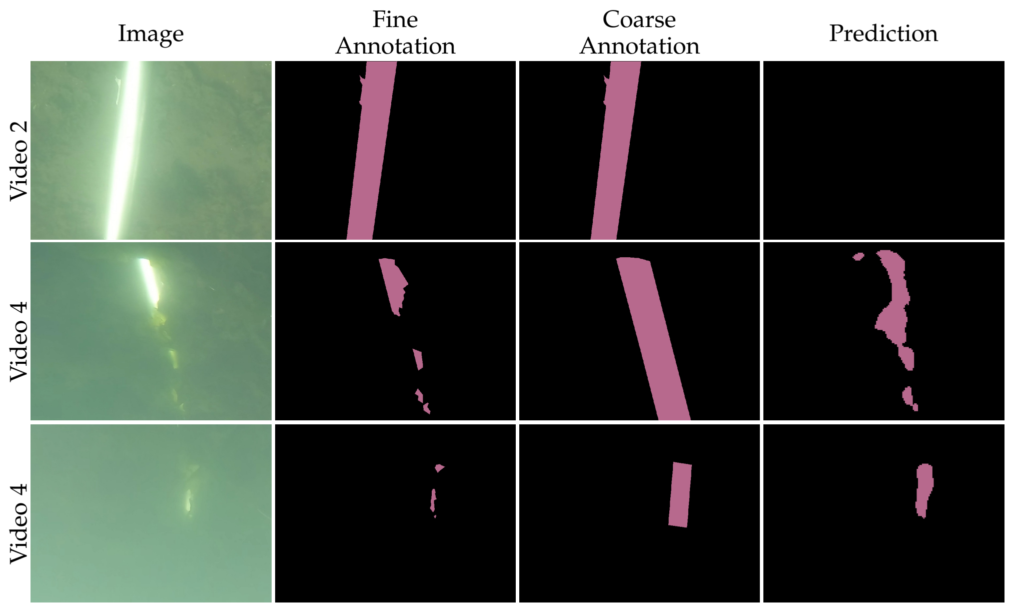


Figure 4.9: Examples of the results obtained with the SegFormer model trained with MarinaPipe.

model, trained with ground truth where only the visible parts of the pipe were annotated, learned to extrapolate the detection to identify regions that may not be perceived by human eyes. This result opens a discussion on the optimal approach to annotate images for training deep learning models: fine or coarse. More experiments and discussions using this dataset are in Chapter 6. Figure 4.9 presents examples of predictions made by the SegFormer model trained with MarinaPipe.

4.4 Discussion and Conclusions

The development of the three datasets discussed above aimed to decrease the gap in the literature of publicly available underwater datasets focused on industrial applications for training deep learning models. The links to download the datasets are available in the respective repositories in the REMARO GitHub:

- github.com/remaro-network/MIMIR-UW (MIMIR);
- github.com/remaro-network/SubPipe-dataset (SubPipe);
- github.com/remaro-network/MarinaPipe-dataset (MarinaPipe).

The experiments performed in this chapter show that MIMIR is a valuable dataset for studying models' generalization, especially when dealing with varying light conditions. We also observed that, while the images are highly realistic, there remains a gap between the simulation and the real world. Training models with data augmentation helps reduce the gap between models trained with synthetic and real images, but it does not fully solve it. At the same time, MIMIR is a dataset with patterns that are much closer to images collected in underwater survey than other datasets that are commonly used for pre-training models, such as ImageNet. This makes MIMIR particularly beneficial for pre-training models, which can later be fine-tuned with a small number of real-world images to eliminate the sim-to-real gap, as demonstrated in Chapter 6.

The performance of the models trained to segment the simple pipeline shape in SubPipe highlights the challenges of dealing with low-quality underwater images. These challenges are even more pronounced in MarinaPipe, which contains more blurred images and the additional problem of occlusion by algae, making the images difficult to annotate correctly when preparing the dataset. Given the low performance of SegFormer in the pipeline class, MarinaPipe could be considered an open challenge to the underwater computer vision research community. Due to difficulties in annotating MarinaPipe in the most beneficial way to train deep learning models, it could be studied under a incompletely annotated datasets perspective, such as the work developed by Lv *et al.* [105]. The impacts of labeling choices (coarse or fine) are further investigate in Chapter 6. Future work on MarinaPipe is to label more images, which may potentially improve the performance of models trained on this dataset.

Contributions Attribution MIMIR and SubPipe are collaborative projects. The MIMIR dataset was idealized and developed by Álvarez-Tuñón [154]. My contribution to this project consisted of designing and conducting the experiments explained in Sect. 4.1, which analyzes the challenges presented by this dataset and its potential benefits for industrial applications for the task of segmenting pipeline from background in camera images. Similar to MIMIR, SubPipe is a multitask dataset. My contribution to this project was to manually label the RGB camera images for image segmentation and training deep learning models using this data, as described in Sect. 4.2.

Chapter 5

Active Learning

This chapter describes the research that was developed using active learning, and that was published in the paper in Appendix E. In real industrial applications, it is common to have limited financial and human resources for labeling data. Active learning aims to select a small subset of data that can be used for training a model achieving comparable results to those achieved when training with the entire dataset, and saving human efforts for manual labeling. It is usually a much more efficient strategy for selecting data for annotating than random selection.

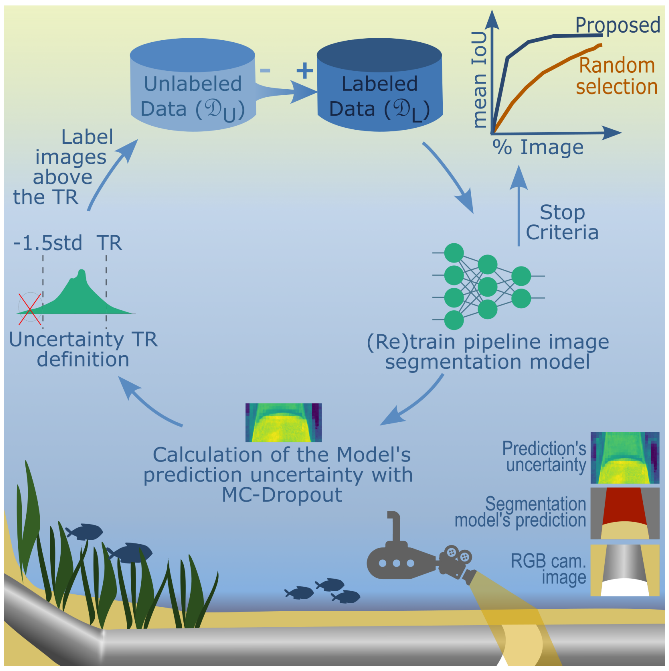

Unlabeled Data ($\mathcal{D}_U$)
-
+
Labeled Data ($\mathcal{D}_L$)
mean IoU
Proposed
Random selection
% Image
Stop Criteria
Label images above the TR
-1.5std
TR
Uncertainty TR definition
(Re)train pipeline image segmentation model
Calculation of the Model's prediction uncertainty with MC-Dropout
Prediction's uncertainty
Segmentation model's prediction
RGB cam. image

Figure 5.1: Our active learning process: After pre-training the model with a small set of randomly chosen images, we start selecting images with uncertainty above a threshold TR for retraining the model. We discard images with an uncertainty below the mean minus 1.5 standard deviations. The signs '$+$' and '$-$' indicate that the newly selected images are removed from D_U and added to D_L.

In this thesis we investigate the use of active learning in the underwater domain. We use a large dataset of underwater RGB camera images recorded during pipeline inspections. These images are semantically similar, but vary in quality due to factors like low and non-uniform illumination, color degradation, low contrast, blurring, and hazing [27, 28, 53]. Labeling these images requires specialized expertise and constitutes a significant cost for the growing but still small underwater automation industry. Selecting key images representing the different image qualities, reduces the required amount of training data, resulting in lower cost and human effort for labeling, which is valuable for innovating companies in this sector.

Inspired by research that successfully applied active learning in other domains, such as medical imaging and autonomous driving, we aim to select images based on uncertainty to minimize the volume of data requiring annotation. Our hypotheses is that active learning can effectively identify key images for training the model, including the different image qualities, and thus saving annotation time while achieving better results than with random selection. We aim to use a state-of-the-art deep learning model that has dropout layers, and train it with active learning, using MC-dropout for selecting images for annotating.

In our study, we investigated the use of epistemic uncertainty for image selection in the field of computer vision applications, cf. Fig. 5.1, more specifically to enhance the capabilities of autonomous robotic systems. We employed two distinct datasets, one for street view image segmentation, and the other for segmenting images captured during underwater robotics missions. The street view dataset was employed for allowing reproducibility. To the best of our knowledge, this was the first research published in a paper applying active learning for real underwater images.

5.1 Methodology

We employed two deep learning models for segmentation, DenseNet and HyperSeg. DenseNet [160], initially developed for classification, was later extended for segmentation [161]. We chose DenseNet due to its success in accessing epistemic uncertainty with MC-Dropout [38].

HyperSeg is a state-of-the-art model based on the U-Net architecture [162]. In the decoder, it utilizes dynamic weights, that are generated based on both the input image and the spatial location. We hypothesize that it is an appropriate choice, because of its high meanIoU performance, high fps rate, and the option to use dropout.

Deep Active Learning Framework. In a real-world scenario the images are labeled after being queried. To respect this, we can not use the labels for image selection. Our active learning process begins by randomly selecting a constant $P\%$ of the images from both the training and validation sets in the first iteration. The model is then trained with these images. The epistemic uncertainty is calculated as the mutual information using MC-dropout, Eq. (2.3), which generates a value for each pixel, and the uncertainty for each image (EU_{img}) is then calculated as the average uncertainty over the pixels, such as in Eq. (2.4). The next step is to calculate the thresholds TR_t and TR_v (Fig. 5.1) used to decide which images from the unlabeled training and validation datasets, respectively, should be selected for labeling. These thresholds are calculated using Eq. (2.5). The images with EU_{img} above the respective thresholds, TR_t and TR_v, are selected for labeling. The model is then retrained with the new and previously selected images.

To accelerate the selection process, images with uncertainty values below 1.5 standard deviations below the mean are excluded from future selection. Since the model's predictions for these samples are relatively certain, using them to retrain the model is unlikely to improve performance. The value of 1.5 was empirically chosen and should be further studied in the future.

5.2 Results

This section discusses the impact of applying active learning on the datasets *CamVid* [142, 143], and *pipeline*.

5.2.1 Datasets

CamVid [142, 143] is a street view dataset for segmentation, which contains video frames captured from the viewpoint of a driving car. Labels for 32 hierarchical classes are provided from which we use 11: 'sky', 'building', 'column-pole', 'road', 'sidewalk', 'tree', 'sign-symbol', 'fence', 'car', 'pedestrian', and 'bicyclist'. This dataset contains a total of 701 images.

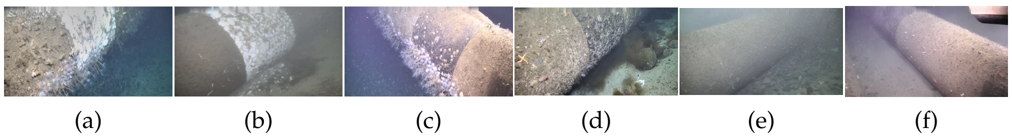


Figure 5.2: Example of the pipeline dataset: (a-b) an anode and pipeline, (c) a pipeline and a field joint, (d) a field joint, boulders and pipeline, (e) a pipeline alone, and (f) a pipeline, field joint, and part of a vehicle in the top right (best viewed in color).

The underwater dataset *pipeline* was provided by EIVA's customers and contains images from pipeline surveys. Due to security reasons,[1] the data is not publicly available. The entire dataset comprises 64,920 video frames recorded during surveys of infrastructure. The dataset has five classes: 'background', 'pipeline', 'field joint', 'anode', and 'boulder-and-survey vehicle', which appear in 99.9%, 77.1%, 20.3%, 21.0% and 15.5% of the training and validation images, respectively. The percentages do not sum to 100% because an image can contain more than one class. Figure 5.2 shows several examples of images in this dataset.

We divide the two datasets into three subsets of images: training, validation, and testing. For the experiments with active learning, new images are iteratively selected from the pools of unlabeled training and validation images and added to the respective pools of labeled images, cf. Fig. 5.3. For the test subset, the labels of all images have been used since the beginning, and this subset of images remains unchanged during all experiments and iterations. The flow diagram in this image shows how the images are moved and applied in each active learning iteration.

5.2.2 Results of the Active Learning Experiment with the CamVid Dataset

For the experiment using DenseNet, we start training the model with 10% of the images. With the trained model, the uncertainties of the images are calculated. Then, the thresholds TR_t and TR_v are defined with Eq. (2.5) for selecting training and validation images with uncertainty higher than 1 standard deviation above the average for labeling. The model is than re-trained using the new labeled images, and the uncertainties are calculated again. This repetition of training the model, calculating the uncertainty and TR, and selecting images for labeling is repeated for some iterations. For the CamVid experiments, we stop if very few

[1]E.g. https://en.wikipedia.org/wiki/2022_Nord_Stream_pipeline_sabotage

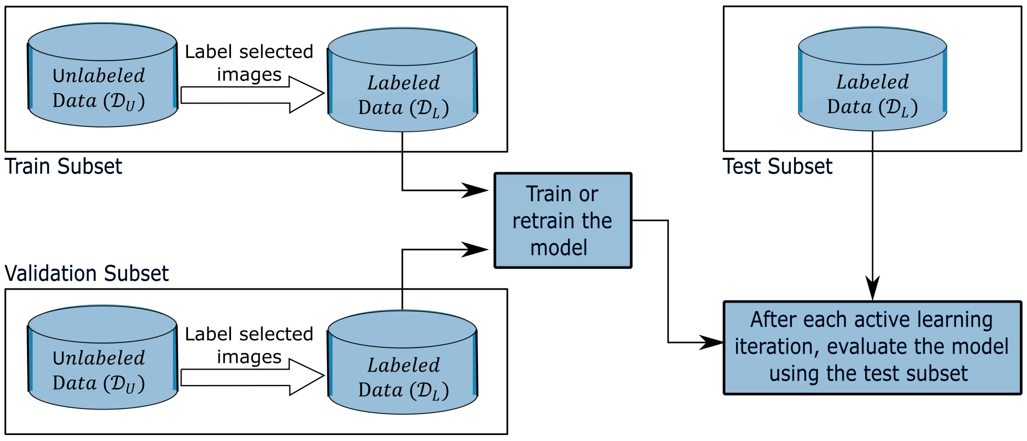

Unlabeled Data ($\mathcal{D}_U$)
Label selected images
Labeled Data ($\mathcal{D}_L$)
Train Subset
Labeled Data ($\mathcal{D}_L$)
Test Subset
Train or retrain the model
Validation Subset
Unlabeled Data ($\mathcal{D}_U$)
Label selected images
Labeled Data ($\mathcal{D}_L$)
After each active learning iteration, evaluate the model using the test subset

Figure 5.3: The two datasets used, CamVid and pipeline, were divided into training, validation, and test subsets. For the active learning experiments, a few images from the training and validation subsets are initially labeled. At each iteration of the experiments, more images are selected, without replacement, from the respective pools of unlabeled images. The selected images are labeled and moved to the pools of labeled data. The entire test subset has labels since the beginning of the experiment.

images with uncertainty above the threshold are left. For evaluation purposes, a second model is trained using randomly selected images. The number of randomly selected images is equal the number of images selected based on uncertainty. Figure 5.4(a) presents the outcomes of the experiment on the CamVid dataset using DenseNet. The model's meanIoU stabilizes around 59% after iteration 12, using approximately 40% of the training data and slightly over 50% of the validation data, which represents around 41% of the data when the weighted mean is calculated for both subsets as shown in Fig. 5.4(a). After that, only one new image was selected for training and one for validation, indicating that additional images do not contribute significantly to the model's knowledge. Comparatively, training the model with the entire dataset yielded only a slightly higher meanIoU of 60.22% but at more than double the cost. Furthermore, the model trained with uncertainty-selected images consistently outperformed the one trained with random images.

Additionally to DenseNet [38], we investigate the performance of HyperSeg, for which we start by training the model with 10% of the images, and select new images for labeling if their uncertainty is higher than 1.5 standard deviations above the average. The model trained with active learning outperformed the model trained with random images until around 25% of the images were queried, Fig. 5.4(b). After 25% of the images were selected, the performance of the models trained

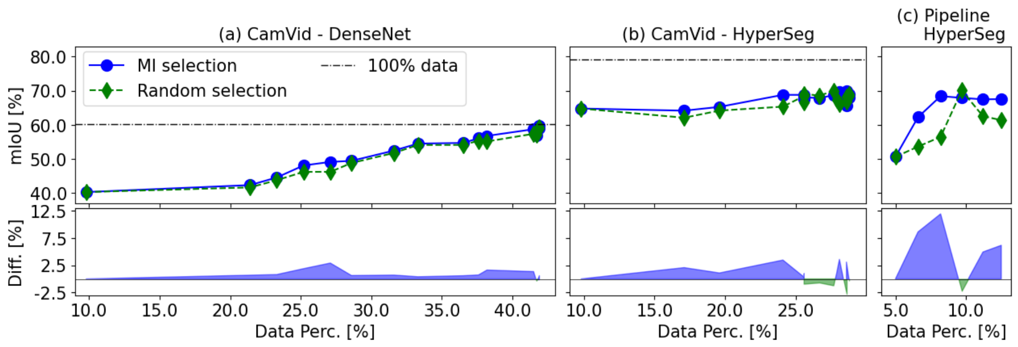


Figure 5.4: Results of the active learning experiments. The bottom graphs show the difference in meanIoU between the model trained with active learning vs. trained with random images, where positive values means active learning prevails.

with images queried with uncertainty and randomly were very similar and stabilized around 69.0%. A possible reason for this behavior is that HyperSeg model has excellent generalization capabilities, requiring fewer data to achieve a stable meanIoU close to the performance obtained with the entire dataset. Finally, Fig. 5.5 and Tbl. 5.1 show results obtained with the models from the last iterations of the DenseNet and HyperSeg experiments. Table 5.1 also shows the performance of the models trained with the entire dataset.

In summary, applying active learning to CamVid confirmed the effectiveness of the active learning framework. It further reinforced that when data follows a consistent pattern, adding more samples does not necessarily improve the model's performance. As shown in Fig. 5.4(a-b), the meanIoU gain for DenseNet from the beginning to the end of the experiment is much more expressive than for HyperSeg. Even though more studies should be performed to analyze this behavior, we hypothesize that it regards the models' structure. HyperSeg apparently has a huge capability of generalization and does not require as much data as DenseNet to achieve the best performance possible.

Figure 5.6 compares our results with three recent deep active learning methods. Semi supervised semantic segmentation for active learning (S4AL) achieved around 61.4% meanIoU training on 13.8% of the data, which is 97% of the performance obtained with the entire dataset [47]. Similarly, Manifold Embedding-based Active Learning (MEAL) achieved 59.6% meanIoU, which is 81.6% of the overall performance, with just 5% of the images [46]. However, these frameworks queried patches of images instead of the whole image.

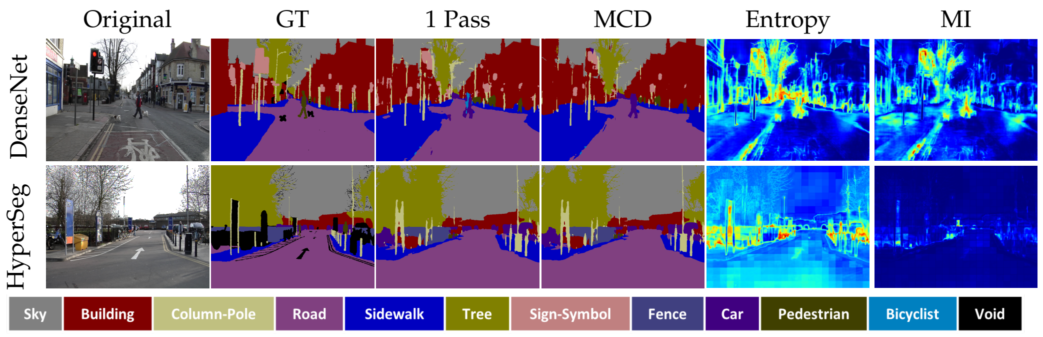


Figure 5.5: Test segmentation results for the models trained with CamVid. GT is the ground truth. MCD is the average of the results obtained with MC-dropout. For the entropy and the mutual information (MI) plots, the warmer colors represent higher values. The entropy and MI heatmaps are normalized per experiment. (best viewed in color.)

Difficulty-Aware Active Learning (DEAL) used image-level querying, and achieved 61.64% meanIoU, which is about 95% of the whole dataset performance, using 40.0% of the data [45]. Using DenseNet, we achieved 97.9% of the whole dataset's results with 41.9% of the data. With HyperSeg, we obtained 87.1% performance using only 28.9% of the data. As HyperSeg is a state-of-the-art model tailored for CamVid, the meanIoU is higher than for the other approaches. Notice also that the same framework requires a different percentage of data to obtain results close to the result obtained with 100% of the data when different model architectures are used, as we demonstrated using DenseNet and HyperSeg. Finally, when analyzing the results, it is important to remember that S4AL, DEAL, and MEAL are much more complex frameworks than ours. S4AL uses a teacher-student architecture for allowing semi-supervised training using pseudo labels. MEAL uses uniform manifold approximations and projection (UMAP) to learn a low dimensional embedding representation of the encoder output and uses K-Means++ to find the

Table 5.1: Results for the CamVid test subset. mIoU refers to meanIoU.

Experim.	% Data	Sky	Build.	Column	Road	Sidew.	Tree	Sign	Fence	Car	Pedes.	Bicyc.	mIoU
DenseNet	100	88.8	74.3	30.8	89.5	77.0	71.3	33.1	32.2	68.1	51.6	45.6	60.2
	41.9	91.9	77.6	28.2	92.0	77.6	74.1	26.5	26.0	70.0	47.5	37.3	59.0
HyperSeg	100	94.5	92.9	50.3	97.4	88.5	86.4	35.3	70.8	94.4	73.8	86.8	79.2
	28.8	85.8	84.5	38.5	94.3	7.2	79.9	47.7	55.5	88.7	46.5	61.0	69.0

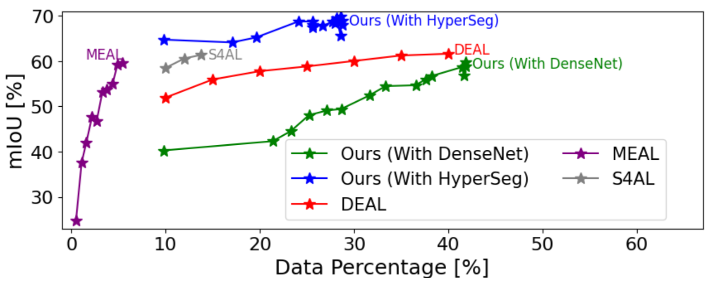


Figure 5.6: Comparison of methods on CamVid. The values for DEAL, MEAL, and S4AL were manually extracted from the graphs in the original papers, hence are approximate. For each framework, the markers indicate when new images were labeled, and the models were (re-)trained in the respective original papers. (Best viewed online in color.)

most representative between the most informative images sampled with entropy. DEAL adds to the CNN structure a probability attention module for learning semantic difficulty maps. Our framework consists of taking an out-of-the-box model without modifications and allowing dropout during the inference phase for using the MC-dropout and calculate the epistemic uncertainty.

5.2.3 Results of the Active Learning Experiment with the Pipeline Dataset

The active learning process takes longer on the pipeline dataset, because it is larger than CamVid. We chose to run this experiment with HyperSeg, as it required less data and fewer iterations in the CamVid experiment, while achieving significantly higher meanIoU than DenseNet.

In this experiment we start by training the model with 5% of the data, and select new images for labeling if their uncertainty is higher than 1.5 standard devaiations abome the average. Figure 5.4(c) presents the results obtained. For this experiment, we stop the active learning framework if the meanIoU does not improve significantly anymore. The model trained with uncertainty-based selection outperforms the baseline model trained with randomly selected images. The final meanIoU of the active learning model is 6.17% higher than the baseline. The model trained with active learning was much more stable, presenting better and increasing performance across iterations. The last considerable improvement in the performance of the model trained with active learning was from iteration 2 to iteration 3. The model trained with randomly selected images achieved a comparable performance at iteration 4, which, however, seems to be a lucky outlier. Hereafter, the performance drops significantly in

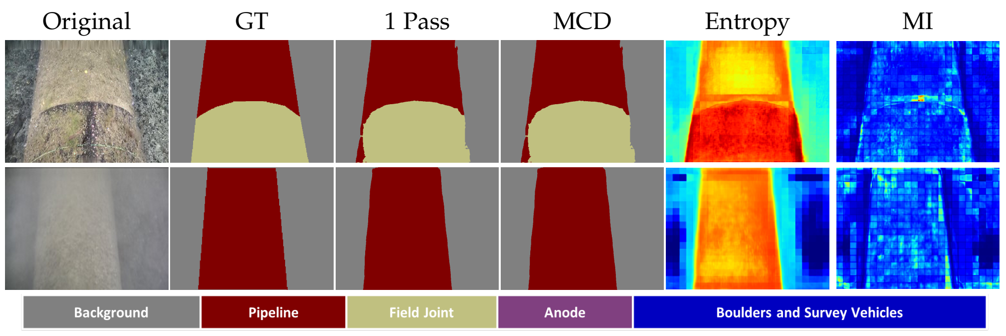


Figure 5.7: Example results for pipeline test images. For the entropy and the mutual information plots, the warmer colors represent higher values (best viewed in color).

the following iteration. Considering the amount of images labeled in iterations 3 and 4, the model trained with active learning achieved a comparable performance to the model trained with randomly selected images with 15.9% fewer images used for training the model, which means 15.9% less annotation work. Table 5.2 presents the meanIoU for the test dataset for the final iteration models with uncertainty-based selection and with the baseline random selection. Note that for the most underrepresented class (boulder and survey vehicle) the model trained with uncertainty-based selection presents the most significant performance gain compared to the model trained with random images. Figure 5.7 showcases two examples of predictions for test images using the resulting model from the last iteration of this experiment. The entropy plots show higher values for the pipeline and the field joint, the relatively illegible classes, suggesting that using entropy as an acquisition function could yield good results too.

In Fig. 5.4 the difference of meanIoU between the models trained with images selected based on uncertainty and the ones trained with randomly selected images is more significant in the pipeline experiment than in the CamVid experiments. A possible reason is that the pipeline

Table 5.2: mIoU for the pipeline test dataset, for the models trained in the final iteration using 12.5% of the data. The percentages over headings indicate the amount of images containing each class in the entire training and validation datasets.

	99.9%	77.1%	20.3%	21.0%	15.5%	
Selection Criteria	Backg.	Pipe	F. Joint	Anode	B.&S. V.	mIoU
Uncertainty	97.0	84.6	66.3	31.1	58.6	67.5
Random (baseline)	96.2	80.0	61.3	29.0	40.3	61.4

dataset is much bigger and unbalanced than CamVid, making the selection of images more critical and challenging.

5.2.4 Repeatability.

Learning depends on the initial set of images chosen for the first iteration. To test repeatability we reran the CamVid experiment with HyperSeg several times. We use CamVid because it is a smaller dataset than pipeline, allowing to run the experiments faster. The original setup for CamVid with HyperSeg used $S = 1.5$ for thresholds in Eq. (2.5), for selecting images with uncertainty higher than $S = 1.5$ standard deviations above the average for annotating. For the repeatability test, we used both $S = 1.5$ and $S = 0.5$, the latter selecting more images in each iteration. The top plot in Fig. 5.8 shows meanIoU results after each iteration for active learning and baseline random-selection models, grouping runs starting with different initial sets of images using four colors. As can be observed in the middle plot, active learning always prevails with $S = 1.5$. The bottom graph shows that for $S = 0.5$ the performance gain was smaller, and in the experiment number 2, in pink, random selection performed better. We hypothesize that as CamVid is a very small dataset and HyperSeg has a strong generalization capacity, when more data is selected the benefit of the uncertainty selection gets much lower. It remains an open question whether retraining the models from scratch at each iteration would prevent them from getting stuck in local minima yielding better performance. Notice that we reported in the previous paragraphs the performance for the experiment 1, in blue. The other experiments presented better meanIoU%. Experiment 4, in yellow, for example, achieved 75.7% meanIoU, 95.6% of the performance with the entire dataset, training on 21.7% of the data.

5.3 Discussion

We demonstrated the effectiveness of active learning with epistemic uncertainty in an underwater infrastructure inspection task, using HyperSeg, a five-class dataset of more than fifty thousand images, and mutual information as the epistemic uncertainty measure. The HyperSeg structure did not need to be modified, making this method easy to implement. Using active learning for selecting the training images resulted in a model with 6.17% better meanIoU than a baseline model trained with the same number of random images. The model trained

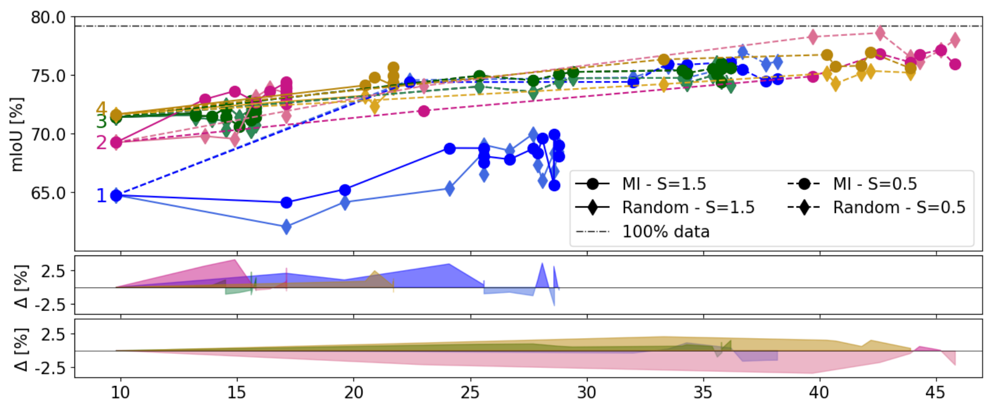


Figure 5.8: Study of repeatability of results using the CamVid dataset and the model HyperSeg. Top: Four colors group run for different random initial sets of images. Below: mIoU differences Δ between the active learning and the baseline random selection models for $S = 1.5$ (middle) and $S = 0.5$ (bottom). Positive Δ means active learning prevails. (Best viewed online in color.)

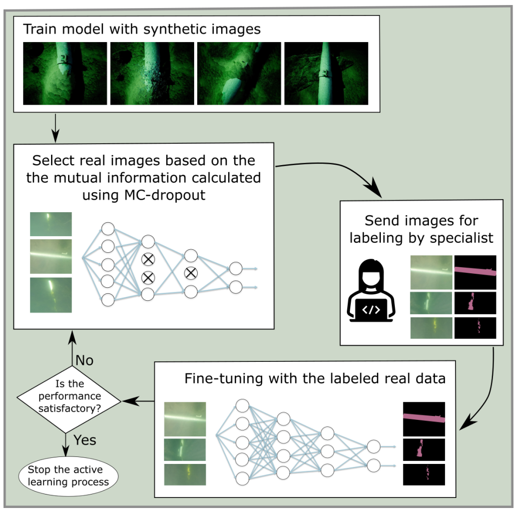
with active learning achieved 67.5% meanIoU using only 12.5% of the available underwater dataset for training and validation. We observed that in the second iteration, the images queried attempted to compensate for the less represented classes in the dataset and the classes with lower performance in the previous iteration. This indicates that the approach helps with unbalanced datasets.

Our experiment on the CamVid dataset, a small street view dataset with 11 semantic classes, suggested that the framework may not provide a significant advantage for small datasets. However, in our experiment with the underwater dataset, the uncertainty-based selection required 15% fewer images than the random selection to achieve comparable results, as can be observed in the iterations 3 and 4 in Fig. 5.4. A possible reason for this results is that the underwater datasets is much bigger than CamVid dataset, the class imbalance is very accentuated (see Tbl. 5.2), and it varies a lot in image quality.

A similar approach as the one used in this chapter will be used in Chapter 6 to bridge, with minimal effort, the gap between models trained with synthetic data and models trained with real data.

Chapter 6

Sim-to-Real



Train model with synthetic images
Select real images based on the the mutual information calculated using MC-dropout
Send images for labeling by specialist
Fine-tuning with the labeled real data
Is the performance satisfactory?
No
Yes
Stop the active learning process

Figure 6.1: Sim-to-Real with Active Learning.

Labeling images for every new task or data pattern a model needs to learn is a significant time bottleneck in real-world applications. Moreover, acquiring the necessary data for training the models can be challenging. A possible solution is to train the models with simulated images. Although modern simulators are highly advanced and close to real scenarios, a gap still exists between their patterns. To bridge this gap, active learning can be applied to select a small amount of real images that are useful to be annotated and used for fine-tuning a model pre-trained with synthetic images [163], similarly to the approach used in Chapter 5).

In this research, published in the paper in Appendix D, we study the sim-to-real gap between MIMIR (Sect. 4.1), which is a synthetic dataset, and MarinaPipe (Sect. 4.3), which contains images collected in a physical environment. The issues posed by real underwater images, such as non-uniform illumination, low contrast, color degradation, and motion blur, are challenging to realistically mimic in synthetic images. To overcome this gap, our approach consisted on training a segmentation model on MIMIR, then selecting MarinaPipe images with hight uncertainty for annotation, and fine-tuning the model with the selected images (see Fig. 6.1). To the best of our knowledge, the paper in Appendix D was the first published study regarding the sim-to-real gap for RGB underwater images using active learning.

6.1 Methodology

In this section, we present the methodology and details for our experiments with pipeline image segmentation.

Pre-Training with Synthetic Data The first step of the experiments was to train a segmentation model on the synthetic dataset MIMIR, Sect. 4.1. From the several environments that MIMIR contains, we used the images from SandPipe, which contains a single pipeline on the ocean floor. We used the images collected by the bottom-facing camera in the simulated underwater vehicle, similar to the position of the camera that collected the real dataset MarinaPipe, 4.3.

We chose to use the visual transformer SegFormer [158] for segmentation, modifying it to include dropout layers in order to be able to use the methodology of Chapter 5 in this study. Since the final goal was to use the trained model with real underwater images, several randomized augmentations were performed to reduce the overfitting to MIMIR, smoothing the transition from simulation to reality, Fig. 6.2. The augmentation techniques employed include:

- Perturbation of the RGB channels and the value and saturation in the HSV color space;
- Resizing, cropping and flipping;
- Conversion to grayscale;
- Addition of motion and Gaussian blur.

Fine-Tuning with MarinaPipe In the pre-training phase with synthetic data the model was trained from scratch on MIMIR. After this pre-training the active learning method was applied to select the most relevant images from the real underwater dataset for fine-tuning the model. The method applied is the same explained in Chapter 5. Like previously,

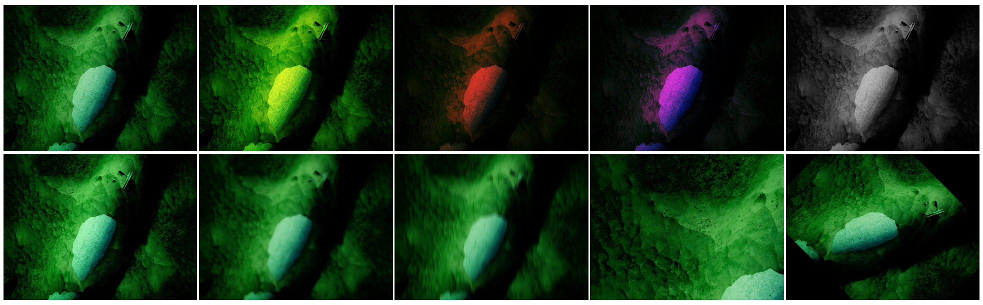


Figure 6.2: Examples of augmentations applied to MIMIR. Top: The most left image is the original; the next three are examples of RGB channels perturbation; and the most right is a conversion to grayscale. Bottom (from left-to-right): Value and saturation perturbation; Gaussian blur addition; motion blur addition; resize with cropping; and rotation. In the images with blur addition, notice how the object contours get weaker and how the screw in the pipeline joint "disappears" (Best viewed online in color with zoom-in.)

the epistemic uncertainty of each prediction was calculated as the mutual information using MC-dropout, and the mean epistemic uncertainty over all pixels of each image EU_{img}, Eq. (2.4), was used as the acquisition function for querying new images. Images with high EU_{img}, above a chosen threshold TR, were selected for fine-tuning the SegFormer model that was pre-trained with MIMIR. As in the previous research, TR was calculated using Eq. (2.5) for querying images with uncertainty higher than S times standard deviations above the mean. In the fine-tuning phase with real data, the decoder and encoder of SegFormer were frozen, and only the head of the model was allowed to train. During both pre-training and fine-tuning, the classes used were background and pipeline.

6.2 Results

The sections below present the results of the experiment. We investigate the influence of data augmentation, freezing and unfreezing the decoder layers, the learning rate values, and the choice of using fine or coarse annotation.

6.2.1 Pre-training with MIMIR

From the SandPipe images (of MIMIR) selected for this study, 90% were reserved for training, 5% for validation, and the other 5% for testing SegFormer. After training, the model obtained 88.80% meanIoU on

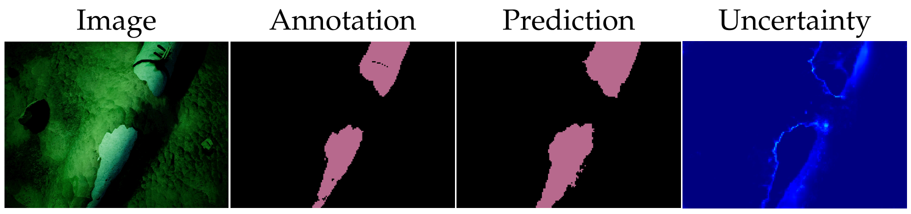


Figure 6.3: An example of prediction on the test dataset from MIMIR using the model trained only on MIMIR. The pipeline and background color are different from Fig. 4.4 because MIMIR has annotation for many classes and we are only using the pipeline class. Everything that is not pipeline is defined as background in this experiments. For the uncertainty plot, calculated as the mutual information, the warmer colors represent higher values.

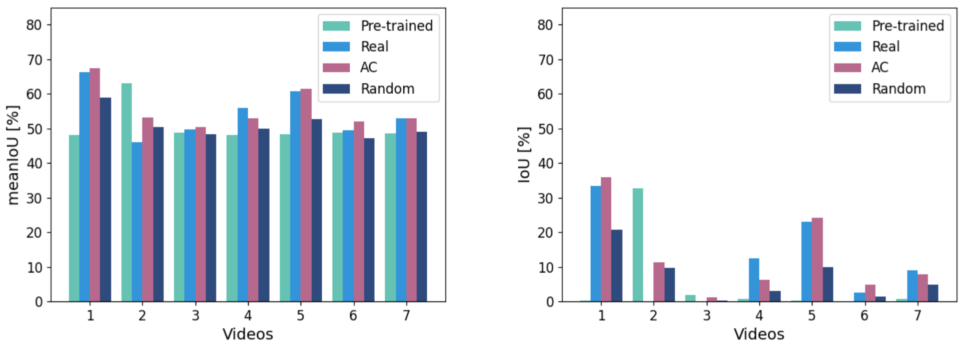
the test dataset, with 81.05% IoU for the pipeline class and 96.56% for the background. Figure 6.3 shows an example of prediction using the pre-trained SegFormer on the MIMIR test subset.

6.2.2 Fine-tuning with real data

The results of the experiment are summarized in Fig. 6.4, where the legends refer to the following set-ups:

- *Pre-trained:* the model trained only on the MIMIR dataset at Sec. 6.2.1.
- *Real:* the model trained in Sect. 4.3 with the real world MarinaPipe dataset. The model was trained with the data from video 1, Tbl. 4.3, and validated with video 7. Only resizing, cropping and flipping augmentations were applied.
- *AC:* the model was pre-trained with MIMIR, model from Sec. 6.2.1, and then fine-tuned with ca. 45% of images selected with active learning from videos 1 and 7 for training and validation, respectively. The fine annotation was used as ground truth.
- *Random:* the model training and fine-tuning was done as in *AC*, however the images were randomly selected, instead of based on uncertainty.

The meanIoU results in Figure 6.4 were calculated using the coarse annotation (see Sect. 4.3) as ground truth. For both *random* and *AC*, the same random augmentation techniques from Sec. 6.1 were applied to the training and validation images used to fine-tune the model.



(a) meanIoU (b) IoU of pipeline

Figure 6.4: Experimental results. (a) meanIoU for the classes pipeline and background; (b) IoU of the pipeline class. Results were calculated using the coarse annotation as ground truth.

When selecting new images with active learning, AC, we used $S = 3.0$ (cf. Eq. (2.5)) for selecting images with uncertainty higher than 3 standard deviations above the mean. This parameter choice resulted in 110 images selected from MarinaPipe for training and 79 for validation. The same number of images were selected for training and validating the *random* model.

We found that fine-tuning the model with real images always helps. Figure 6.4 shows also that active learning (AC) consistently outperforms random selection (Random). Still, the results of the pipeline class leave room for improvement. The pipeline's low IoU may be due to the dataset's complex patterns, which may require more annotated data to improve the model's performance. Even though the IoU for the pipeline is low, notice that the model fine-tuned with active learning (AC) can recognize this object, Fig. 6.5. Observe in this figure, that the model trained only on MIMIR can recognize the pipeline in videos with no occlusion, in accordance with Fig. 6.4, but is much worse than the fine tuned model, when applied to images with occlusion.

6.2.3 Choice of the augmentation, training and structure

In the above, we fine-tune the model with relatively little data. To increase the model's performance, we augmented the images and froze the entire encoder-decoder structure during the fine-tuning. Now, we present an ablation study of the choices made during the fine-tuning phase. All the analyses were performed using the same image frames that were

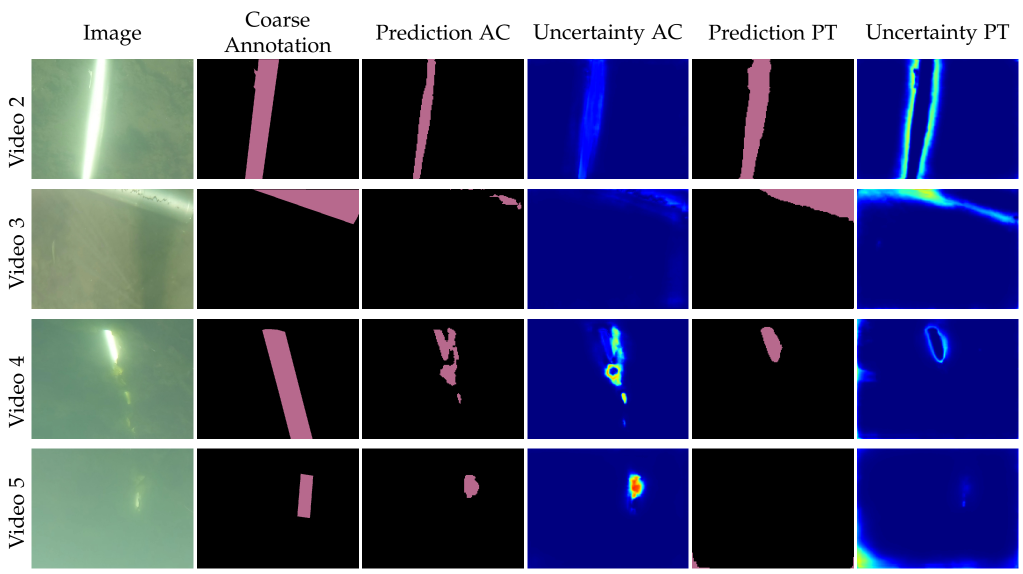


Figure 6.5: Examples of predictions using the model fine-tuned with active learning (*AC*) and the pre-trained model (*PT*), along with the respective mutual information uncertainties. Warmer colors represent higher uncertainty and reflect the models' difficulty when predicting pipelines.

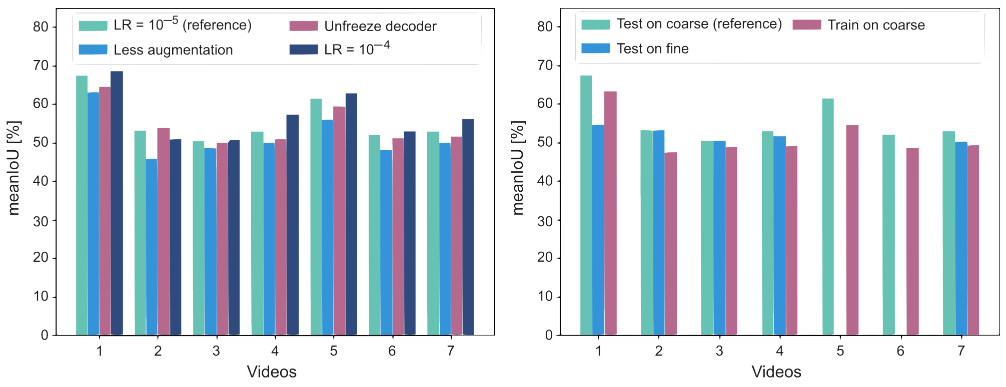
used for fine-tuning and validating the model *AC* in Sec. 6.2.2. In each experiment, we do everything as in Sect. 6.2.2 and apply a single change in each test.

Data augmentation Instead of using all the augmentations listed in Sect. 6.1, we now only apply resizing, cropping, and flipping to the data used for fine-tuning the model to test if the augmentations were confusing the model, or if they were helpful.

Freezing the decoder For analyzing the benefits of freezing the decoder during the fine-tuning phase, now only the encoder was frozen, allowing the decoder and the model's head to train.

Learning rate For analyzing the initial learning rate choice during the fine-tuning, we test setting the initial value to 10^{-4} instead of the 10^{-5} used in Sect. 6.2.2. Both the encoder and decoder were frozen, and only the head was allowed to train.

Figure 6.6(a) shows the results. Decreasing the amount of augmentation and unfreezing the decoder decreased the model's performance. Increas-



(a) Augmentation, learning rate and freezing decoder. (b) Coarse vs. fine annotation

Figure 6.6: Results of the ablation study. 'reference' is the *AC* (active learning) model trained in Sect. 6.2.2. Test on coarse and test on fine use the same model *AC* model, but one is evaluated using the coarse annotation as in Sect. 6.2.2 and the other using fine annotation. Train on coarse was trained and evaluated with coarse annotation. Notice that videos 5 and 6 do not have the fine annotation for performing the evaluation.

ing the initial learning rate gave slightly better results. This was the only model fine-tuned with an initial learning rate equal to 10^{-4} in this study.

Test with fine annotation As stated in Sec. 6.2.2, the models were fine-tuned using the fine annotation as ground truth. In the mentioned section, the model's performance was evaluated using the coarse annotation. Here we also evaluate the model against the fine annotation. The results in Fig. 6.6 show that the performance is higher when using the coarse annotation as ground truth. Apparently, the model learned how to extrapolate the fine annotation.

Learning with coarse annotation Since evaluating the performance with the coarse annotation gave better results, we wonder if training on coarse annotation would result in a better model. However, as Fig. 6.6(b) shows, the results of training using the coarse annotation were worse. We hypothesize that this is the case because the fine annotation gives a more “precise” label, and the model learns better to differentiate the pipeline from the rest of the image.

6.3 Discussion

The key findings of this section are as follows. Active learning is more efficient in fine tuning the segmentation model previously trained on synthetic images than random image selection. The MarinaPipe is a very difficult dataset. It has a lot of motion blur, uneven illumination, and occlusions, which could be the reason for the pipeline class's low IoU.

In this experiments, SegFormer was trained with MIMIR from scratch and then fine-tuned with MarinaPipe. An interesting next test is to pre-train the model with a larger dataset, such as the vastly used COCO [153] dataset, before using MIMIR and fine tuning on MarinaPipe. It would be interesting to see whether this would result in better IoU for the pipeline class. Even though it was demonstrated that SegFormer can be used with active learning, more tests should be performed to study the best positions to insert the dropout layers in this structure.

In this chapter, we used the methodology from Chapter 5 to leverage predictive uncertainty to adapt a model trained on synthetic data to the image patterns of a dataset recorded in a physical environment using minimal labeling effort. In the next chapter, we aim to analyze the reliability of predictions of a model that uses a minimal number of labeled images to teach a trained model to recognize new classes while keeping the knowledge of old classes.

Chapter 7

Generalized Few-Shot Segmentation

Adapting deep learning models to recognize new classes with limited examples can be important for some industrial applications with classes of rare occurrence, e.g., training a model to detect abnormalities with few available samples [164, 165]. Few-shot learning addresses this need, but suffers from the drawback of forgetting previously learned classes. Generalized and incremental few-shot learning have been developed to enable learning new classes while keeping the knowledge about old ones. However, it is important to remember that deep learning models with different architectures are often overconfident [166, 38]. There is no reason to think that generalized few-shot learning models would not suffer from the same issue [167]. Therefore, the predictive uncertainty [38] of generalized few-shot learning models should also be analyzed. We hypothesize that predictive uncertainty can be used to select the prediction about which the model has enough knowledge, increasing the reliability of the predictions. Furthermore, we propose that it is possible to analyze whether a few-shot segmentation model is well trained for new classes by analyzing the variances in the embedding space used to generate the predictions. In this chapter, we explore the results of using entropy and mutual information to evaluate the predictive uncertainty, generating insights into the model reliability.

7.1 Methodology

In this research, we aim not only to assess the predictive uncertainty of the model, but also to investigate an approach to evaluate whether the

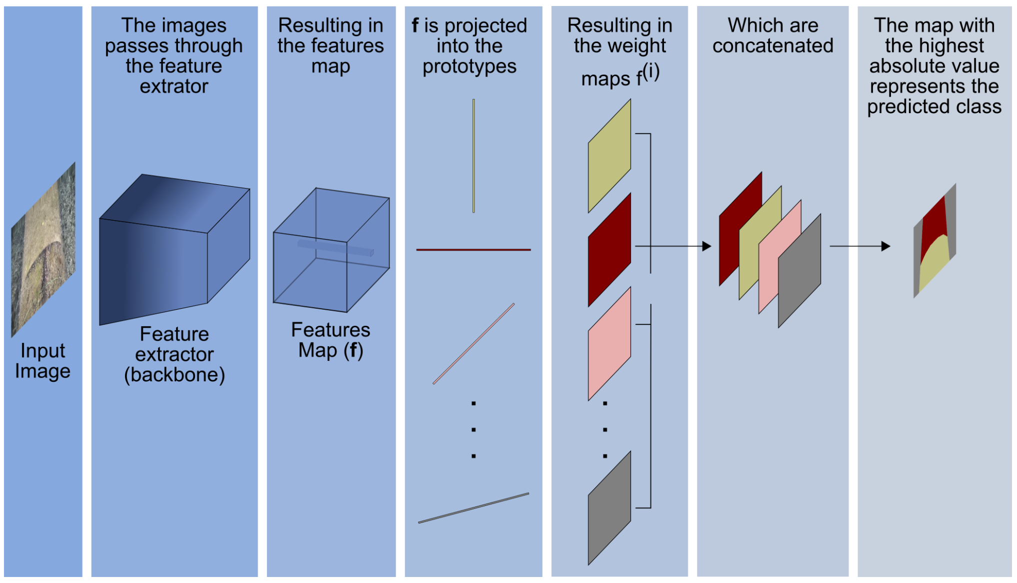

Input Image
The images passes through the feature extrator
Feature extractor (backbone)
Resulting in the features map
Features Map (**f**)
f is projected into the prototypes
Resulting in the weight maps $f^{(i)}$
Which are concatenated
The map with the highest absolute value represents the predicted class

Figure 7.1: The adapted POP structure used in this thesis.

model was trained with sufficient samples (shots) of new classes to effectively learn the patterns required to detect those classes. We hypothesize that the key for evaluating whether a model has appropriately learned the patterns of the new classes lies in analyzing the embedding space. For this reason, we choose to work with prototypical neural networks, which generate a prototype to represent each class, and use these prototypes to analyze the embedding space generated by the model. We further hypothesize that this same approach could be applied to other generalized or incremental few-shot learning models that use prototypes to aid the prediction.

The methodology described here has two goals:

- Define an uncertainty threshold using the training data, which can be used to evaluate the predictions reliability during inference;
- Evaluate whether the model effectively learned the patterns of the new classes by analyzing the variances in the embedding space.

Model and Training Details The model chosen for these experiments was POP [168], a state-of-the-art model for generalized few-shot segmentation that uses prototypes. This model consists of a backbone for extracting a feature map, a module that decomposes the feature map into several unit bases and weight maps, and a classifier. Figure 7.1 presents the adapted version of POP, used in this thesis. The backbone comprises a pyramid pooling module [169] on top of an ImageNet pre-trained ResNet50. The output of the pyramid pooling module is a feature map $\mathbf{f} \in \mathbb{R}^{(HxW)xC}$ that is decomposed as:

$$\mathbf{f} = \sum_{i=1}^{r} f^{(i)} * \mathbf{u}_i = \underbrace{\sum_{i=1}^{M} f^{(i)} * \mathbf{u}_i}_{A} + \underbrace{\sum_{i=M+1}^{r} f^{(i)} * \mathbf{u}_i}_{B} \tag{7.1}$$

with:

$$\mathbf{u}_i * \mathbf{u}_j^T = 0, \quad for \quad i \neq j \tag{7.2}$$

where r is the rank of the matrix $\mathbf{f}$, M is the number of classes that the model is trained to segment, $\mathbf{u}_i \in \mathbb{R}^{1xC}$ are unit bases, and $f^{(i)} \in \mathbb{R}^{(HxW)x1}$ are weight maps calculated as:

$$f^{(i)} = \mathbf{f} * \mathbf{u}_i^T \tag{7.3}$$

The unit vectors in the part A of Eq. (7.1) are the prototypes learned during the model's training to represent M defined classes. The part B of Eq. (7.1) is calculated as the residual of $\mathbf{f}$ after deducting A and is responsible for detecting the background of the images.

In the original POP model, a set of convolutional layers is added after the decomposition module to form the classifier. In this research, the classifier was removed to increase the importance of the prototypes in the pixels classifications. The output corresponding to each class i is $f^{(i)}$. Additionally, dropout layers were strategically inserted after the backbone, before the module responsible for decomposing the feature map, for enabling the use of MC-dropout, described in Chapter 2.

The loss used for training the model was $L = L_{ce} + \lambda L_{orth}$, where λ is a constant for weighting the orthogonal loss L_{orth}, and L_{ce} is the cross entropy loss. L_{orth} is used for enforcing the orthogonality of the prototypes and is defined as:

$$\frac{1}{r'(r'-1)} \sum_{i \neq j} |\mathbf{u}_i * \mathbf{u}_j^T| \tag{7.4}$$

where r' is the number of classes and $r' < r$.

The model is initially trained with many samples of the base classes. Then, it is updated using N samples (shots) of each new class that needs to be learned plus N samples of each base class. During the updating of new classes, the backbone and the prototypes of the base classes are frozen.

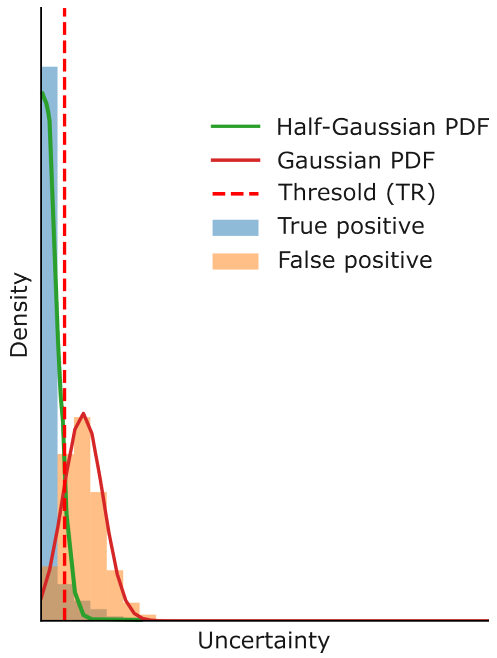
Uncertainty in the Embedding Space In this experiment, we aim to calculate the uncertainty in the distance between each vector in the feature map $\mathbf{f}$, extracted after the dropout layers, and the prototypes. To achieve this, we forward pass the same input image several times through the model. Given that $\mathbf{f} \in \mathbb{R}^{(HxW)xC}$, we compute the distance between each vector $f^{(1x1)xC}$ within $\mathbf{f}$ and each prototype. When the same input is forward passed T times through the model with the dropout layers enabled, each vector $f^{(1x1)xC}$ generates T distance values relative to each prototype. As a result, the uncertainty of each vector $f^{(1x1)xC}$ can be represented by M standard deviation values, one for each prototype, calculated from the T distances.

Uncertainty in the predictions The uncertainty in the predictions is calculated in a similar manner to the approach in chapters 5 and 6. For a more comprehensive analysis, the uncertainty is calculated with three metrics: (1) mutual information using Eq. (2.3), (2) entropy MC-dropout using Eq. (2.1), and (3) entropy without MC-dropout.

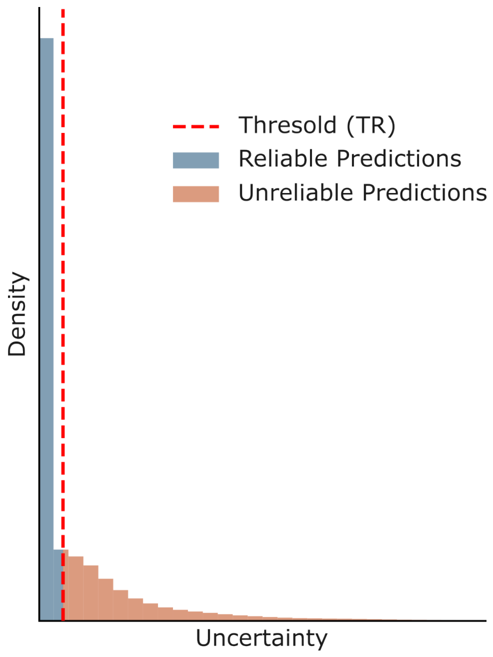
For each of these metrics, and for each class learned by the model, we draw the distributions of the true positive predictions and the false positive predictions, considering only the images used during the new class updating phase. An uncertainty threshold TR is defined for each class as the intersection point of these distributions, as exemplified in Fig. 7.2. During inference, the pixels predicted as class c with uncertainty above the threshold $TR_{c,metric}$ are considered unreliable, while those with uncertainty below the threshold are considered reliable.

7.2 Results

This section discusses the results obtained when calculating the uncertainties both in the embedding space and in the model's predictions. In these experiments, the 'training data' refers to the images used for new class updating, which contains the same number of shots (samples) of base classes as the number chosen for new classes.



(a) Training data

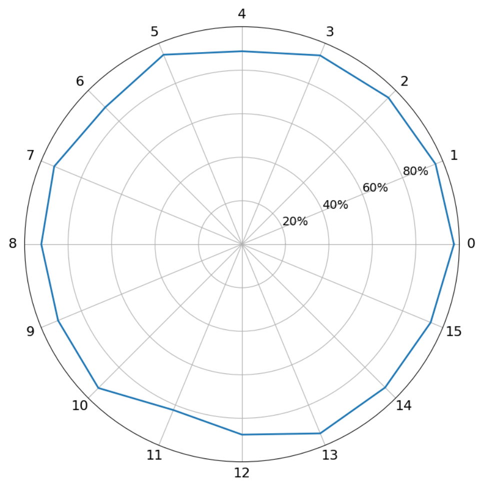


(b) Test data

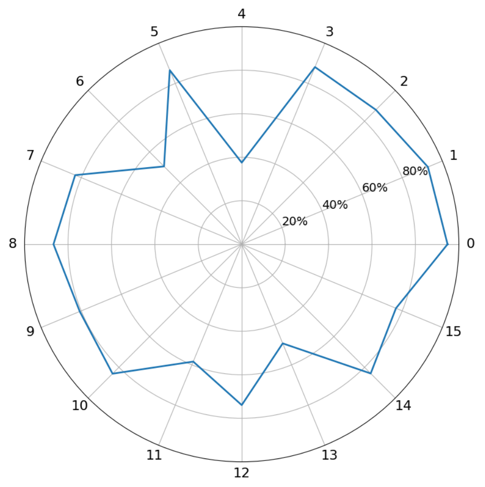
Figure 7.2: Example of uncertainty threshold definition for pixels predicted by the model as class c.

Dataset The dataset used was PASCAL-5i [141], which is a version of PASCAL VOC [106] prepared for training few shot-segmentation models. It contains 4 folds, each with 5 classes. Three of these folds are used for training base classes and one for training new classes, resulting in 15 base classes and 5 new classes. We chose as base classes: ’bus’, ’car’, ’cat’, ’chair’, ’cow’, ’dining table’, ’dog’, ’horse’, ’motorbike’, ’person’, ’potted Plant’, ’sheep’, ’sofa’, ’train’, and ’TV/monitor’. For new classes we used: ’airplane’, ’bicycle’, ’bird’, ’boat’, and ’bottle’.

Model Performance Figures 7.4 to 7.6 report the IoU of the model trained with 1, 5 and 10 samples per new class, respectively. The model performed well in the training data, with only a few classes with IoU slightly lower than 80%. This indicates that the model is effectively learning the patterns present in the training data. However, the important question is whether the model overfits to the limited images used during training or if it performs well on testing images. When comparing with the results of the model trained only for base classes, Fig. 7.3, the performance of the model for the classes in the test dataset does not change significantly in the few-shot models, indicating that significant



(a) IoU of the training set



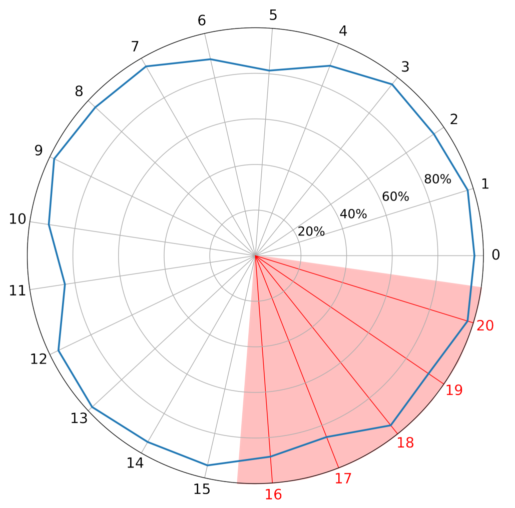
(b) IoU of the test set

Figure 7.3: Performance of the base model trained with enough data. Class '0' is background.

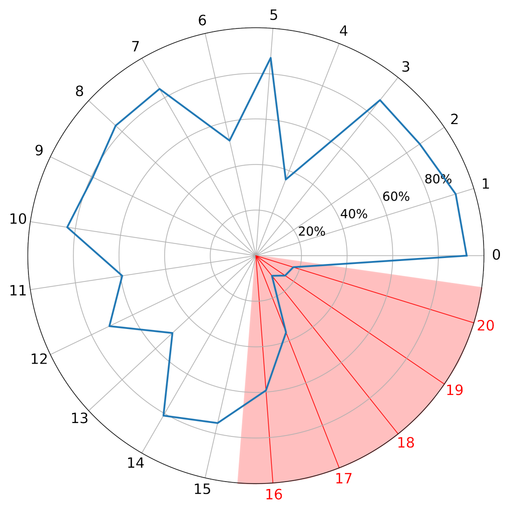
forgetting is not occurring. However, the performance of the new classes is much lower. For the 1-shot model, class 16 presents the highest performance between new classes, with IoU equal to 59.28% , while class 18 has the lowest performance, with an IoU of only 11.29%. As expected, the performance of new classes increases as more shots are used for updating the model. For the 10-shots model, class 17 presented the worst performance, exhibiting an IoU of 29.65%.

Predictive Uncertainty While it is important that a model performs well, a model that does not perform exceptionally can be still acceptable if it is possible to evaluate which predictions are trustworthy. As an attempt to discard unreliable prediction, we define uncertainty thresholds as explained in Sect. 7.1 and illustrated in Fig. 7.2.

Taking class 18 as an example, in the 1-shot model, the reliability analysis using mutual information only considers 4.94% of the predictions reliable. Consequently, the zero percent of IoU for this class, presented in Fig. 7.7, is less worrying, as the model is essentially recommending to discard all the data predicted as class 18. Observe that between the three metrics used, the entropy without MC-dropout saves more predictions as reliable, but it is also more overconfident than the others. For the same class 18, the entropy without MC-dropout considered 16.93% of the predictions reliable, even thought the IoU of the reliable predictions is 9.78%; less than the 11.29% of IoU presented by the model for this



(a) IoU of the training set



(b) IoU of the test set

Figure 7.4: 1-shot model performance. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.

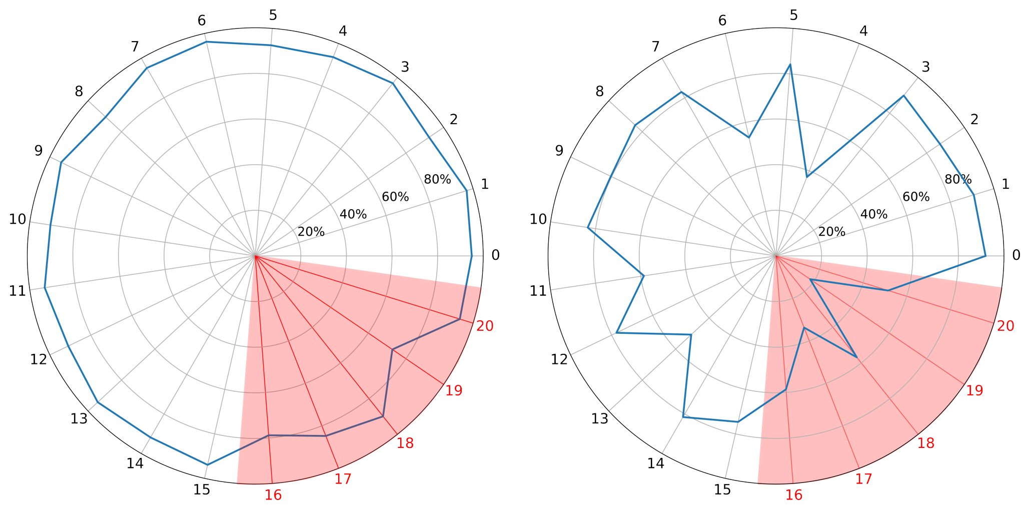
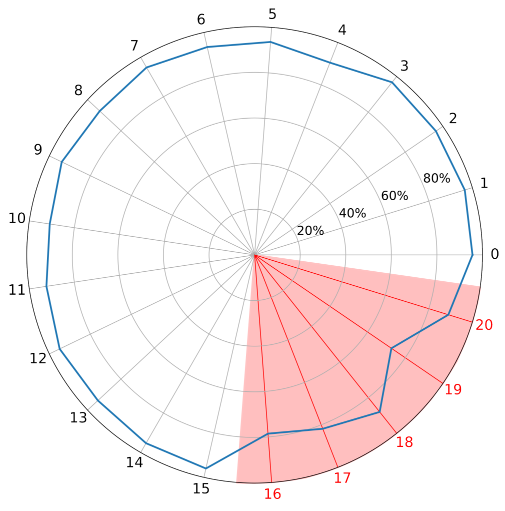
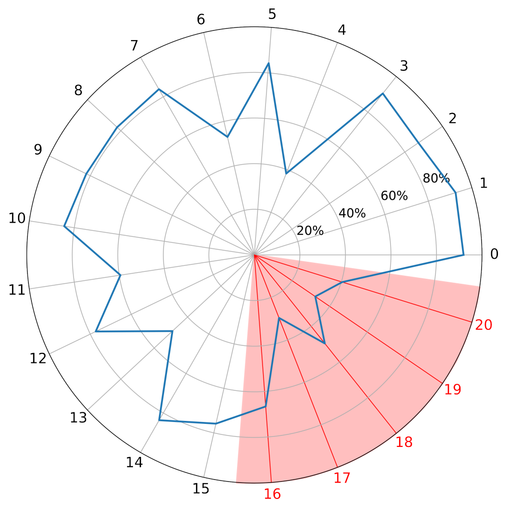


Figure 7.5: 5-shots model performance. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.



(a) IoU of the training set

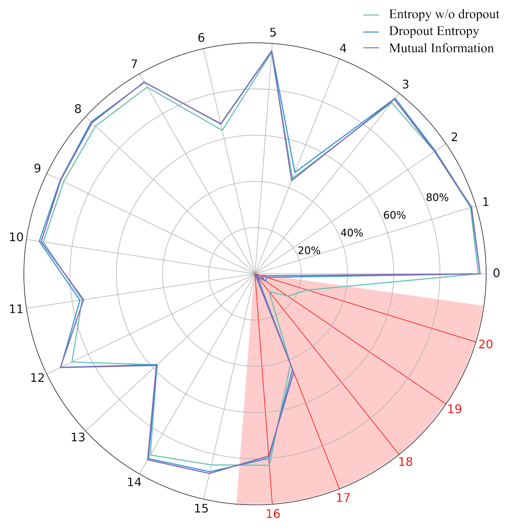


(b) IoU of the test set

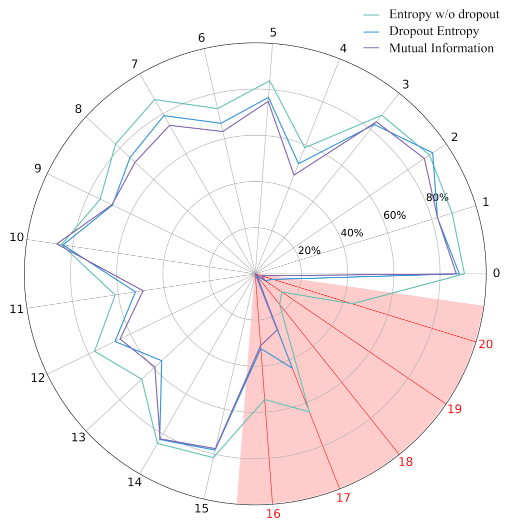
Figure 7.6: 10-shots model performance. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.

class (Fig. 7.4). The results indicate that calculating the entropy with MC-dropout is more reliable than not using it.

As observed in figures 7.4 to 7.6, the models' performance improves as more shots are used for the new classes updating. Moreover, the uncertainty analysis also becomes more effective, as seen in figures 7.8 and 7.9. A possible reason that contributes for the improvement of uncertainty analysis is that more samples are used for calculating the uncertainty thresholds, helping to better capture patterns suitable for the test dataset. For the 10-shots model, the new class with the lowest performance after the uncertainty analyzes with mutual information is class 17, with 45.89% of IoU in the 56.84% reliable predictions. The performance increase after uncertainty analyzes in the 10-shots model for classes 18, 19, and 20 is particularly impressive when comparing to the performance in Fig. 7.6. For class 18, for example, the IoU increases from 49.65% to over 85% after the analysis using mutual information defined 36.91% of the data as reliable. Despite the reliability upgrade with the uncertainty analysis, the approach still needs to be improved for critical application. For instance, the reliable predictions of classes 4 and 17 present around 45% of IoU for the 10-shots model, showing that this method is not perfect.

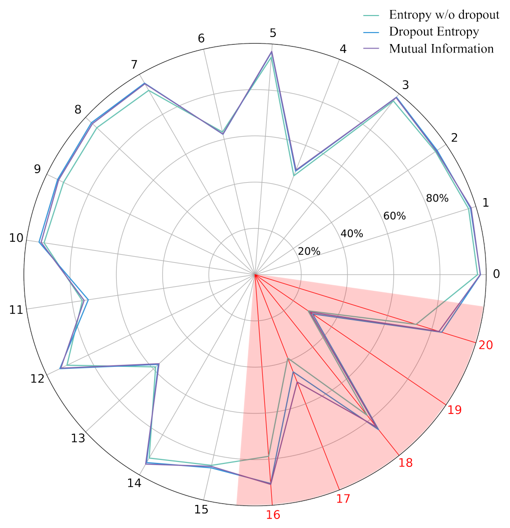


(a) IoU of the reliable data

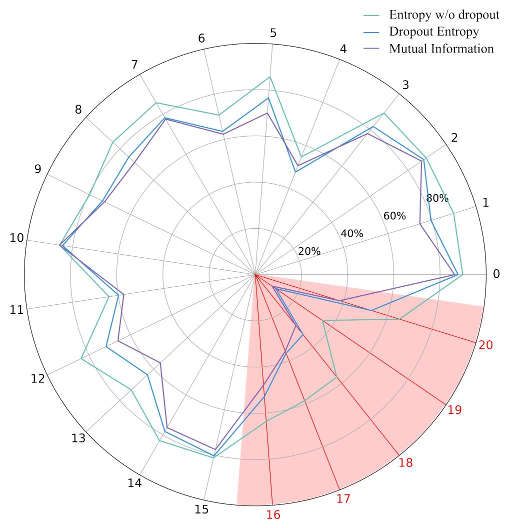


(b) Percentage of reliable data

Figure 7.7: 1-shot model performance after the uncertainty analysis. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.

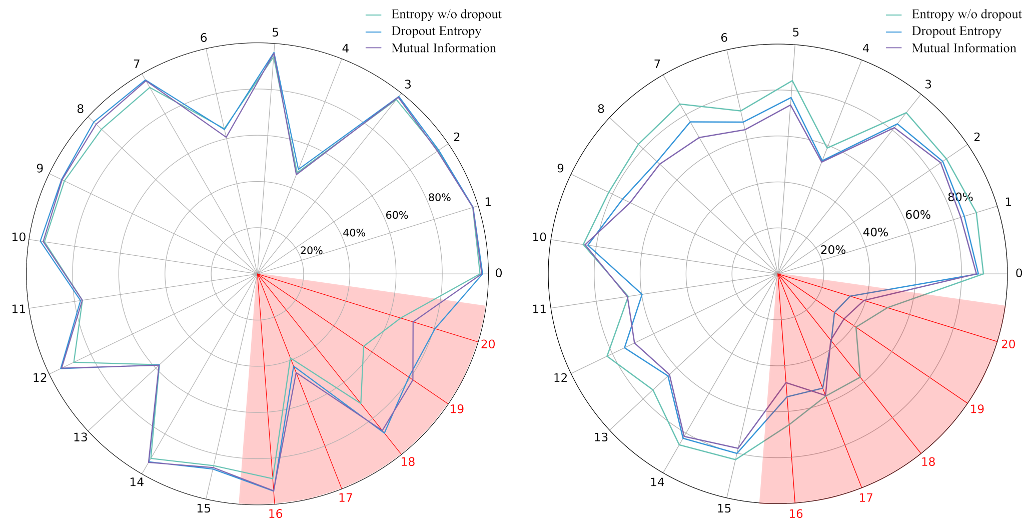


(a) IoU of the reliable data



(b) Percentage of reliable data

Figure 7.8: 5-shots model performance after the uncertainty analysis. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.

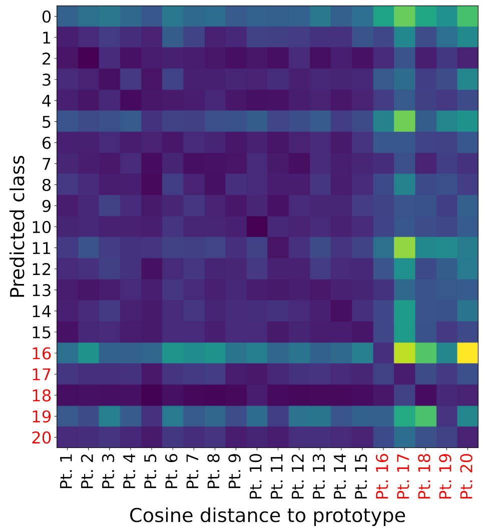


(a) IoU of the reliable data (b) Percentage of reliable data

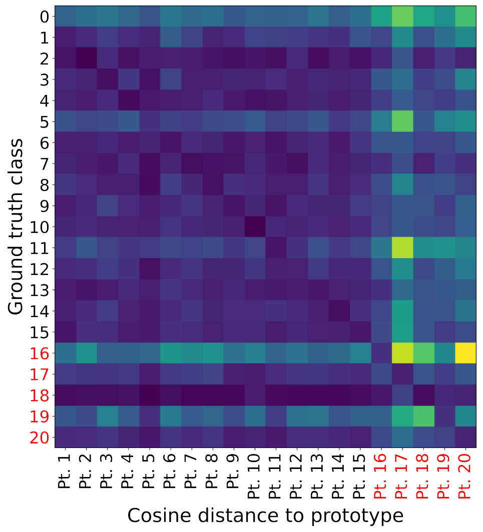
Figure 7.9: 10-shots model performance after the uncertainty analysis. Class 0 is the background, classes 1 to 15 are base classes, and classes 16 to 20 are new classes.

Uncertainty in the Embedding Space Finally, we analyze the uncertainties in the embedding space. Figures 7.10 to 7.12 show the average standard deviation of the cosine distances of the vectors in the feature map $\mathbf{f}$ relatively to the $M = 20$ prototypes when applying MC-dropout with T forward passes. Figure 7.10(a) shows the average of the standard deviations for vectors closer to the same prototype (that will be the predicted class), while Fig. 7.10(b) shows the average of standard deviations for vectors that should be closer to the same prototype (which are the ground truths of the predictions represented by these vectors). The resulting standard deviations are very small, in the order of 10^{-3}. However it is clear from the graphs in Fig. 7.10, generated using the 1-shot model, that the standard deviation of distances to the prototypes representing the new classes (16-20) are higher than those related to base classes. This suggests that the new prototypes may not have learned to properly differentiate the classes they represent from the other classes.

Figures 7.11 and 7.12 used the 5-shot and 10-shot models, respectively, to perform these analyses. Notice that, as the number of shots used for training the model increases, the differences in the standard deviations of the distances to the prototypes of new and old classes diminishes. This indicates that the model becomes less confused by the new prototypes as more samples are used for learning new classes. These analyses were performed using the training images of the class updating phase.



(a) Average standard deviation to prototypes considering the predicted classes



(b) Average standard deviation to prototypes considering the ground truth classes

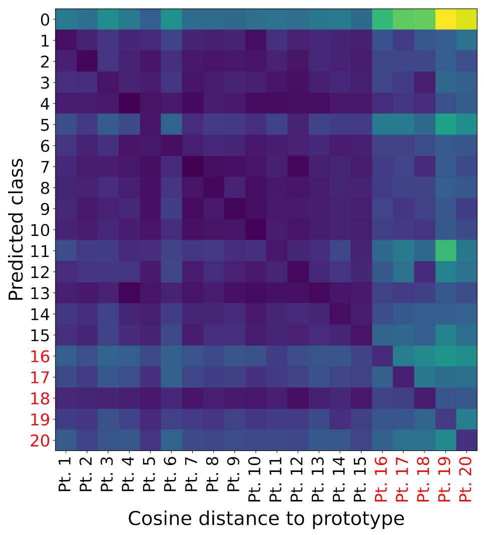
Figure 7.10: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 1-shot model. The experiment was performed using training images. Warmer colors indicate higher values.

The question remain whether this trend is still true when analyzing the test dataset. Figures 7.13 to 7.15 confirms that a similar conclusion can be observed in the test dataset. This indicates that the proposed approach could be used for evaluating if a model is well-trained across all new and base classes.

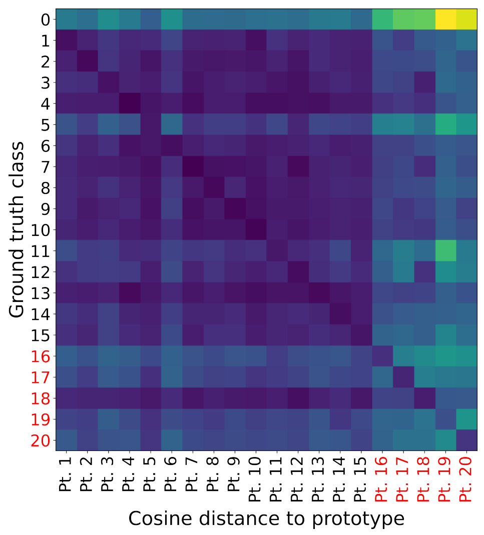
7.2.1 Experiment with an Underwater Dataset

We attempted to perform the same experiment using an underwater dataset.

Underwater Dataset The dataset used belongs to EIVA and was collected by their clients. For security reasons, the dataset cannot be publicly released. This dataset contains seven classes used as base classes and two classes used as new classes. Each base class appears in at least 500 images of the subset of images used during the learning of base classes. The base classes are: 'field joint', 'field joint cover', 'anode', 'pipeline', 'rock dump', 'stabilization mattress', 'survey vehicle'. The new classes are: '*Sarostegia oculata*' (a species of marine invertebrate) and 'stabilization mattress damage'.

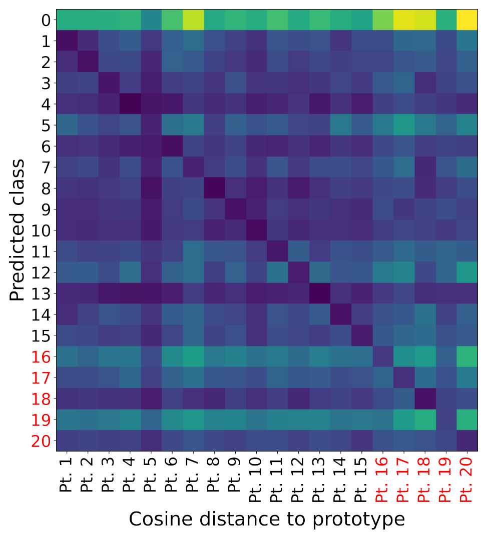


(a) Average standard deviation to prototypes considering the predicted classes

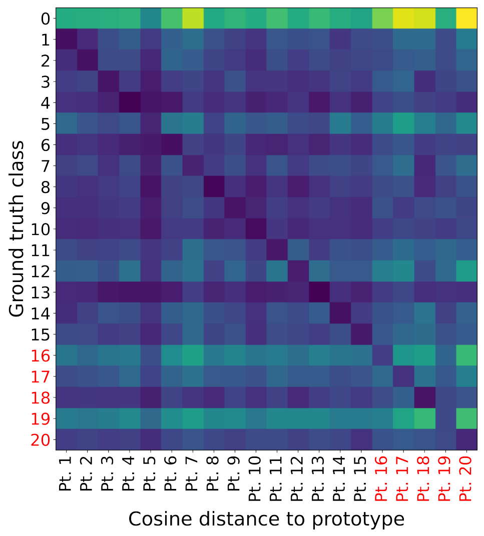


(b) Average standard deviation to prototypes considering the ground truth classes

Figure 7.11: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 5-shots model. The experiment was performed using training images. Warmer colors indicate higher values.



(a) Average standard deviation to prototypes considering the predicted classes



(b) Average standard deviation to prototypes considering the ground truth classes

Figure 7.12: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 10-shots model. The experiment was performed using training images. Warmer colors indicate higher values.

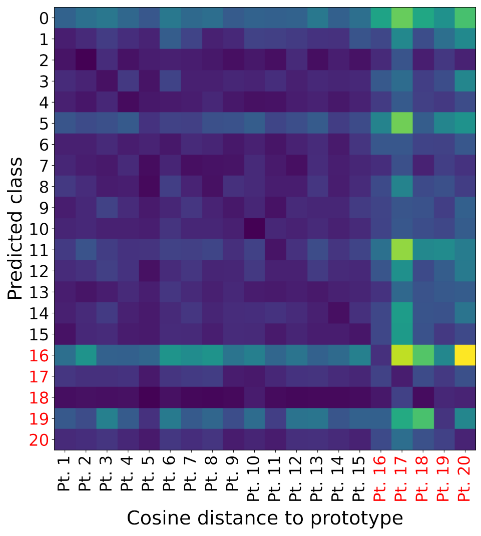


Figure 7.13: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 1-shot model. The experiment was performed using testing images. Warmer colors indicate higher values.

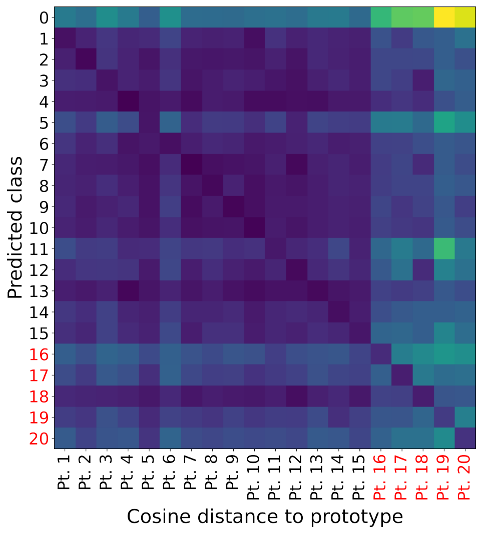


Figure 7.14: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 5-shots model. The experiment was performed using testing images. Warmer colors indicate higher values.

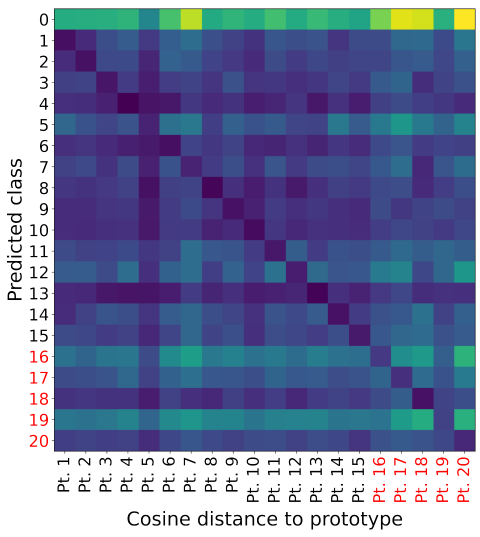
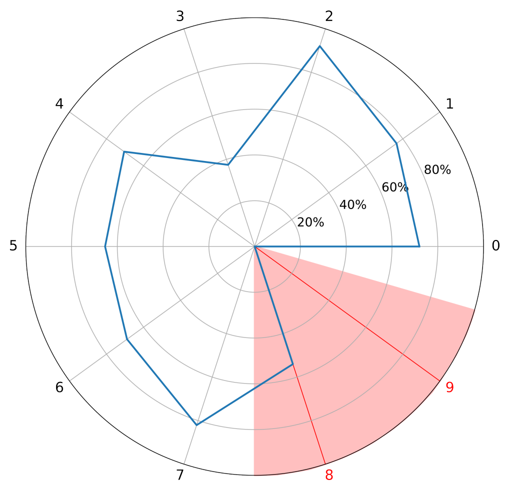


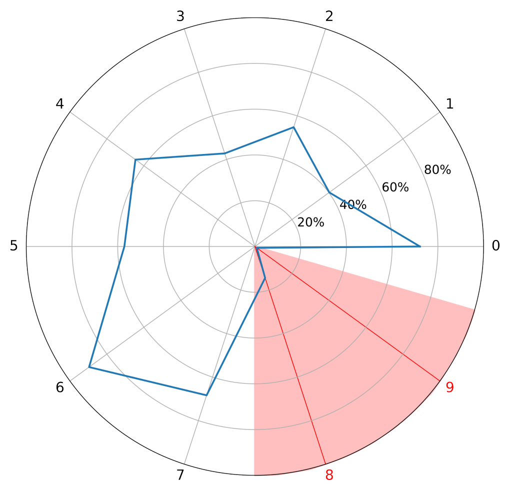
Figure 7.15: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 10-shots model. The experiment was performed using testing images. Warmer colors indicate higher values.

Model Performance The performance of a model trained with 1 shot of each new class is presented in Fig. 7.16. Even for the training data, the performance of the new classes is very low, indicating that the model is not learning the patterns for the new classes in the training dataset. For the test set, the IoU of the class number 9 is 1.0%, while the performance of class 8 is 14.4% of IoU. This poor performance reflects the challenges of performing few-shot learning with this underwater dataset. For instance, '*Sarostegia oculata*' is a class very different from the base classes. At the same time, the 'stabilization mattress damage' is extremely similar to 'rock dump' and the background. This could be the reason for the model not to learn this class pattern. Furthermore, a careful observation of the dataset revealed that some '*Sarostegia oculata*' occurrences are not annotated in the test dataset, which impacts the evaluation performance. We trained a model using 10 shots for each new class, but it still failed to learn the pattern of the 'stabilization mattress damage'.

Predictive Uncertainty Similarly to what occurred for some new classes of the 1-shot model trained with the Pascal-5i dataset, the mutual information considered almost all the predictions for the new classes as unreliable. Considering that the performance for the new classes is extremely low, this result for the uncertainty evaluation is good, alerting



(a) IoU of the training set



(b) IoU of the test set

Figure 7.16: Performance of the 1-shot model trained with the *underwater dataset*. Class 0 is the background, classes 1 to 7 are base classes, and classes 8 and 9 are new classes.

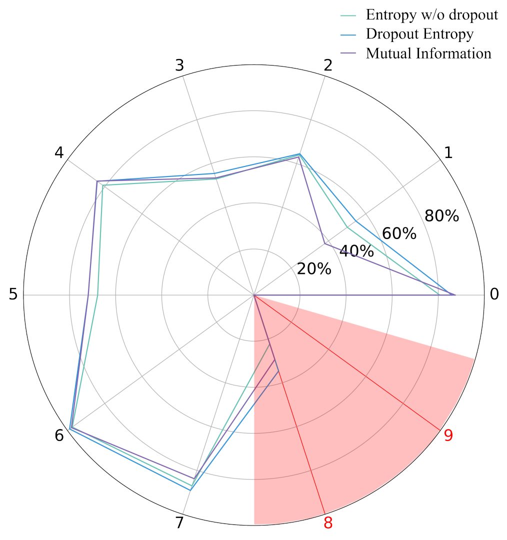
that the model is not well-trained for these classes. The uncertainty analysis increased the performance of the base classes, as observed when comparing figures 7.16 and 7.17.

Uncertainty in the Embedding Space Figure 7.18 presents the evaluation of the uncertainty of the embedding space for the 1-shot model trained with the underwater dataset. Similar to the analyses with the Pascal-5i dataset, the distances to the new prototypes present a higher standard deviation. This indicates that these prototypes are not well defined, and more images should be used for training the new classes.

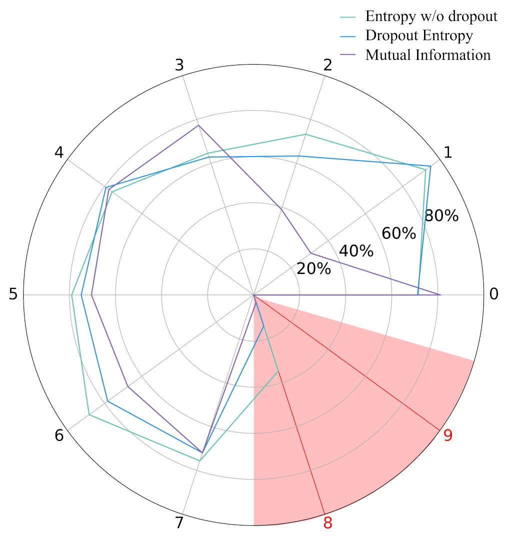
7.3 Discussion

In this chapter, we analyze the feasibility of using predictive uncertainty analysis with MC-dropout to enhance the reliability of a generalized few-shot learning model. Our experiments suggest that mutual information is the most reliable metric for identifying the trustworthy predictions. The results for entropy with and without MC-dropout endorses the hypotheses that using MC-dropout yields more trustworthy results.

The uncertainty thresholds were calculated by fitting half-Gaussian and Gaussian distributions over the uncertainties in the training dataset. However, this distributions may not be the best representations for the

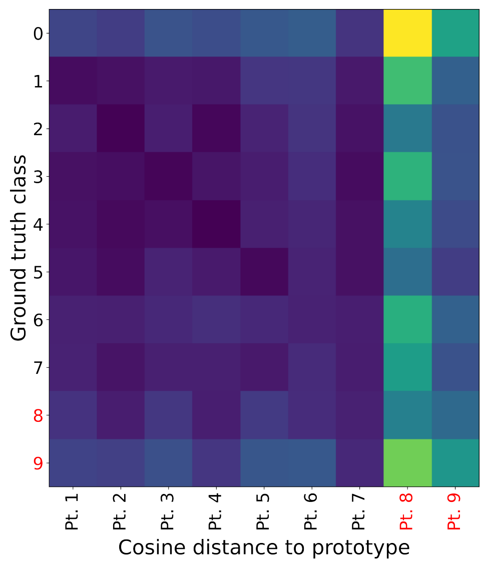


(a) IoU of the reliable data

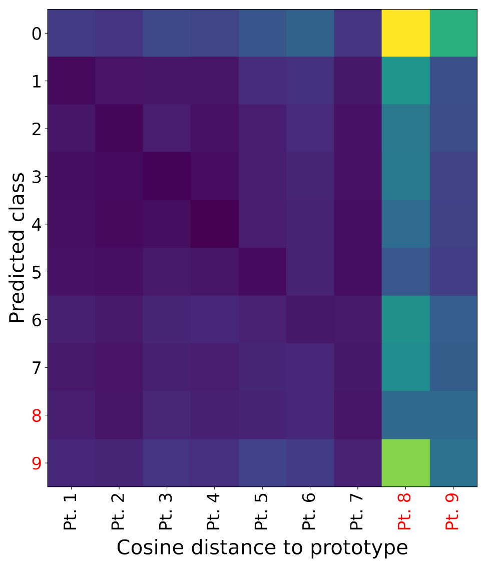


(b) Percentage of reliable data

Figure 7.17: Performance of the 1-shot model trained with the *underwater dataset* after the uncertainty analysis. Class 0 is the background, classes 1 to 7 are base classes, and classes 8 and 9 are new classes.



(a) Average standard deviation to prototypes considering the ground truth classes and the training dataset



(b) Average standard deviation to prototypes considering the predicted classes and the test dataset

Figure 7.18: Average standard deviations of the distances between the vectors in the feature map **f** and the prototypes for the 1-shot model trained with the *underwater dataset*. Warmer colors indicate higher values.

uncertainties, raising the question of whether their use benefited the mutual information over the other metrics. The type of distribution should be further investigated in the future.

The model trained in the experiment that used the underwater dataset exhibited extremely poor performance on the new classes. These results are likely due to the choice of base and new classes. All the base classes consist of man-made objects used in the oil industry or dump of rocks. The two new classes are *'Sarostegia oculata'* and 'stabilization mattress'. While *'Sarostegia oculata'* (a species of marine invertebrate) is significantly different from the base classes, 'stabilization mattress' is visually almost identical to the 'rock dump' class and to the background of the images. Both scenarios proved to be very difficult for updating the model to new classes, with the latter being particularly problematic. Further experiments with a different selection of classes should be explored.

The study in this chapter opens some directions for future research. It would be interesting to analyze whether re-training the model with pseudo-labels generated from the reliable predictions improves the model's performance. Another possible investigation is regarding the use of the uncertainty in the embedding space through the calculation of the standard deviation of cosine distances. Our visual analyses of the heatmaps suggest that the standard deviation was higher for the new classes, and that the differences in the standard deviation for new and base classes decreases as more shots are used for updating the model. This observation leads to the hypotheses that the number of samples used during the new classes update could be gradually increased until the standard deviation of the new classes is comparable to the standard deviation of the base classes. This hypotheses leads to an important question for evaluating models quality: When the standard deviation of new and old classes are comparable, does it mean that the performance of these classes are also comparable?

Chapter 8

Conclusion

In this chapter we discuss the contributions, limitations and possible future research directions regarding the studies conducted in this thesis.

8.1 Summary of Contributions

This thesis focused on reducing the labor-intensive task of labeling datasets and increasing the reliability of deep learning models in the underwater domain. We developed an extensive survey on the state-of-the-art in deep learning models applied to RGB camera image enhancement, classification, segmentation, and object detection in underwater environments. In this survey, we observed that the current publicly available underwater datasets for computer vision are significantly smaller in size and less diverse in class variety than in-air datasets such as ImageNet and Pascal. This limitation is even more evident regarding datasets for industrial applications, where datasets are specially restricted. We also observed that the use of predictive uncertainty for computer vision in the underwater is almost non-existent.

To help address the lack of data, we worked on the development and public release of three underwater datasets for pipeline tracking and segmentation, two of which were designed as multipurpose datasets. Given the interests of this thesis, we worked on the development and evaluation of these datasets for the task of pipeline segmentation in RGB images. The first dataset is the synthetic dataset MIMIR, which is a valuable tool for studying model generalization across different underwater surveys and environments, given that it includes images from several tracks recorded in four simulated environments. The second dataset, SubPipe, consists of data recorded by a physical robot during a real underwater pipeline survey. The images in SubPipe exhibit much more blur than MIMIR images. Despite the simple shape of the straight pipeline in the SubPipe images, the segmentation model SegFormer achieved only 66.4% IoU for the pipeline class. This result highlights the challenges posed by underwater images. The third dataset, MarinaPipe, contains images of pieces of pipe on the seabed of a marina, recorded with a physical robot. MarinaPipe has pipelines covered by algae, and images with visible sun rays, which makes it a challenging dataset to annotate. We release this dataset with both fine and coarse annotation.

In an attempt to reduce the amount of time and costs spent in image annotation for preparing datasets to train deep learning models, we used predictive uncertainty calculated with MC-dropout for smartly selecting images for annotation. In our active learning research, selecting images from an underwater dataset of over 50,000 samples based on uncertainty reduced the labelling effort by 15.9% compared to random selection while achieving a comparable performance of approximately 68% IoU. In our research for bridging the sim-to-real gap with minimal effort, we trained SegFormer on MIMIR for pipeline segmentation and fine-tuned it with a few images from MarinaPipe. Selecting images based on uncertainty resulted in a better meanIoU compared to random selection - on average, the meanIoU was 4.2% higher on images extracted from the 5 videos used for testing. Furthermore, we observed that training the SegFormer model with the fine annotated version of MarinaPipe resulted in better performance than using coarse annotation. The predictions were even able to extrapolate the fine annotation detecting true pipeline regions not annotated as pipe in the original fine annotation.

For a generalized few-shot segmentation problem, which already uses few images for training, we investigated the use of uncertainty evaluation to improve the reliability of the model. We observed that analyzing the uncertainty with mutual information calculated with MC-dropout generated more reliable results than using entropy. Our results also support the literature by demonstrating that calculating the entropy with MC-dropout is more effective for assessing the predictions reliability than without it. We also investigated the uncertainty in the embedding space by analyzing the variations in the distances between vectors from the feature map extracted by the backbone and the class prototypes. This study suggests that these distances become more stable as the number of images used to update the model to new classes increases.

8.2 Shortcomings and Limitations

Given the limited time available for developing research, it is often necessary to restrict the tests performed or to accept certain shortcomings.

For applying the MC-dropout technique for calculating the epistemic uncertainties, we chose segmentation models that already have dropout layers in their structure or added these layers ourselves. Due to time constraints, we were unable to experiment with varying the placement or rate of the dropout layers to choose the best settings. Additionally, we evaluated the ideal number of forward passes necessary for calculating the uncertainty with MC-dropout using the DensNet for segmentation model. We assumed that the same number would be appropriate for other models as well. However, the optimal approach would be to evaluate the ideal number of forward passes for each different structure selected to use.

In the active learning and the sim-to-real projects, we defined two uncertainty thresholds: one for the training dataset and another for the validation dataset. This means that the same threshold was used to evaluate the uncertainty of outputs predicted as different classes. However, for the same model structure, outputs predicting different classes often have different uncertainty distributions, which was not accounted for in these two projects.

In the generalized few-shot segmentation project, a different threshold was applied for each class. These thresholds were defined by determining the intersection point of the two distributions drawn for the uncertainties of true positive and false positive predictions for each class. For the true positives, we used half Gaussian distributions, and for false positives, we used Gaussian distributions. However, these may not be the most appropriate distribution types for modeling this data. Additionally, to speed up the implementation process, the uncertainty thresholds for the base classes were defined using the same number of sample images (shots) as the number used for the new classes. However, the number of images containing base classes is not as limited as the number of images containing new classes. If all the available images with base classes were used for calculating the respective uncertainty thresholds, the results would likely be better.

Finally, MC-dropout was applied to evaluate the epistemic uncertainty. Because it can be applied to models that contain dropout layers without performing modifications in their structure, this is a practical approach.

However, this method requires forward passing the same input multiple times through the model to assess the uncertainty. Consequently, it is not feasible to evaluate the uncertainty of the predictions in real time.

8.3 Future Work

Several investigations could be developed based on the research presented in this thesis, specifically addressing the limitations and shortcomings discussed Sect. 8.2.

As noted in Sect. 8.2, outputs predicting different classes often have different uncertainty levels. In the active learning and the sim-to-real projects, better results would likely be achieved by employing an uncertainty threshold per class. Furthermore, the thresholds in these two projects were defined based on a chosen number S of standard deviations above the average uncertainty value. Higher values of S result in fewer images being selected for labeling. Further investigation should be developed on the ideal number S of standard deviations.

Additionally, in the active learning and the sim-to-real projects, another research path to investigate involves querying images based on the uncertainty of image patches instead of the uncertainty of the entire image. In this thesis, entire images were queried because, given the nature of our images, annotating a patch does not save much time compared to annotating the entire image. However, aggregating the uncertainty of the entire image could hide regions with high uncertainty if the rest of the image has very low uncertainty. This may result in missing the selection of important images containing regions about which the model lacks knowledge and would benefit from train with. Therefore, future work should explore querying images based on patch-level uncertainty and evaluate whether the results are better than selection based on image-level uncertainty.

For the generalized few-shot segmentation research, further studies should be developed to understand the correlation between the variance in the embedding distances and the necessary number of images used during the new class learning. Another possible research direction is to use the reliable predictions as pseudo-labels to improve the model's performance. By employing pseudo-labels for the new classes based on the reliable predictions, it may be possible to train the model with sufficient data for the new classes, potentially resulting in better performance for those classes.

As a final consideration, as explained in Sect. 8.2, it is not feasible to calculate the uncertainty in real time using MC-dropout. This is not an issue for performing offline tasks, such as selecting images during active learning. However, it is a limitation for activities requiring online uncertainty evaluation, such as decision-making or automatically determining whether additional images of a particular location needs to be collected during a survey. For such applications, other methods of uncertainty evaluation, such as the error propagation method, should be explored.

Appendix A

Paper - Exploring the Depths: A Comprehensive Survey of Deep Learning for Underwater Image Processing

Exploring the Depths: A Comprehensive Survey of Deep Learning for Underwater Image Processing

Luiza Ribeiro Marnet, *Member, IEEE*, Yury Brodskiy, Stella Grasshof,
Erdal Kayacan, *Senior Member, IEEE*, and Andrzej Wąsowski, *Member, IEEE*

***Abstract*—Underwater images suffer from problems not encountered in in-air images, such as color degradation, blur, turbidity, uneven illumination, and many small, occluded objects. These characteristics pose unique challenges for image processing. We survey published studies on deep-neural-network-based processing of underwater monocular RGB images, including image classification, segmentation, and object detection. We identify different approaches for improving the performance of deep learning models in underwater images, such as skip connections and feature maps from different layers to deal with objects of various sizes, or transfer and adversarial learning to overcome the challenges of acquiring large datasets in this domain. To address the low quality of underwater images, we cover image enhancement for underwater vision using deep learning models. This includes the effectiveness of the enhancement methods in improving object detection tasks and their implementation costs versus benefits. We list and discuss several publicly available datasets for various deep learning tasks. Finally, we discuss the future directions for underwater image processing research along with the best practices for developing deep learning methods, which can help standardize and thus facilitate the comparison of developed models.**

***Impact Statement*—Deep learning has significantly advanced the field of computer vision, but the best-known models for classification, detection, and segmentation were developed for in-air images. Many attempts have been made to adapt these models to the lower-quality underwater images. This survey provides an overview of this specialized field and analyzes the advantages of enhancing images before applying deep learning models. It also provides an extensive list of publicly available datasets for different underwater-vision tasks using deep learning. We aim to accelerate development in underwater vision applications, as both the summary of the state of the art and the knowledge of available datasets are essential for rapid progress. This survey stands out for its completeness, including a review of publicly available datasets, and enhancement, classification, detection, and segmentation methods in the context of deep learning methods applied to underwater RGB monocular images.**

***Index Terms*—Classification, Deep Learning, Detection, Enhancement, Segmentation, Underwater Camera Images**

I. Introduction

Visual monitoring of underwater environment is a crucial challenge that needs to be solved to enable applications such as underwater species recognition [1], coral detection [2], or monitoring of sea-bed installations, pipelines, and other relevant equipment [3] for marine growth, debris, and potential damage. Monitoring by humans is very expensive and also dangerous, so autonomous computer vision methods for these tasks are needed.

Computer vision is a field that includes capturing [4] and extracting information [4], [5] from scenes. The goal is to automate the processes of interpreting visual data [6], [7], for tasks such as classification [8], [9], object detection [10], [11], and segmentation [12], [13]. Deep learning methods stand out by their ability to extract the relevant information automatically from data [14], [15] without the need for handcrafted features [16]–[18]. Computer vision with deep learning achieved excellent results in some areas, even surpassing human performance [19], [20].

Computer vision is yet to achieve the same success in the underwater environment, where it is challenged both by the difficulty of acquiring the datasets and by the idiosyncratic properties of the underwater imagery. These images suffer from low and non-uniform illumination, color degradation, low contrast, blurring, and hazing [23]–[25]. The scattering and absorption of the light energy cause light attenuation. Scattering also leads to blurring and low contrast [23], while plankton and debris ("marine snow") cause haziness [24]. As light is attenuated

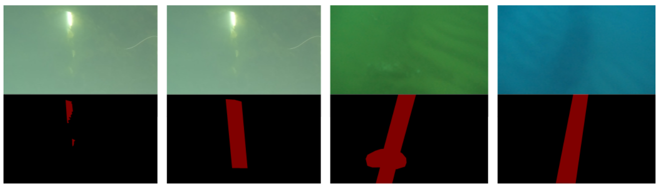


Fig. 1. Two images from the MarinaPipe dataset [21] (left) and two from the SubPipe dataset [22] (right). Below we include the ground truth masks for segmentation. The leftmost is finegrained, the others are coarse annotations.

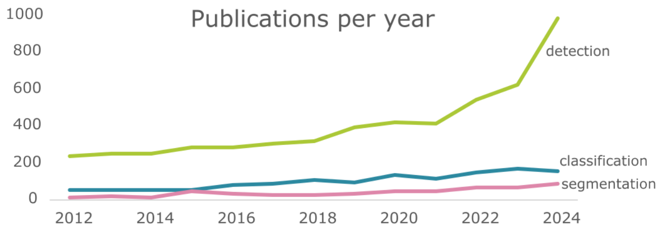


Fig. 2. Publications rate according to Google Scholar, for works that include 'underwater', 'coral', or 'fish' combined with 'classification', 'detection', or 'segmentation' in the title. We excluded entries including 'sonar', 'ultrasound', 'ultrasonic', 'acoustic', and 'acoustics' in the title. For 2024, the number is extrapolated from the amount of publications in the first 5 months of the year.

Partially supported by EU's Framework Program for Research and Innovation Horizon 2020, Grant no. 956200, *Reliable AI for marine robotics (REMARO)*

L. R. Marnet and Y. Brodskiy are with EIVA A/S, 8660 Skanderborg, Denmark (e-mail: {lrm,ybr}@eiva.com).
S. Grasshof is with Computer Science Department, IT University of Copenhagen, 2300 Copenhagen, Denmark (e-mail: stgr@itu.dk)
E. Kayacan is with Automatic Control Group (RAT), Paderborn University, 33098 Paderborn, Germany (e-mail: erdal.kayacan@uni-paderborn.de)
A. Wąsowski is with Computer Science Department, IT University of Copenhagen, 2300 Copenhagen, Denmark (e-mail: wasowski@itu.dk)

differently depending on the wavelength, we observe color deviations. Longer wavelengths are attenuated faster (red), while shorter (blue) are attenuated more slowly [25], giving the underwater images a bluish hue. The visibility ranges from twenty to less than five meters, depending on the water clarity or turbidity. Figure 1 shows some examples of real underwater images. Figure 2 shows the evolution of interest in analyzing underwater images reflected by the number of publications over the years.

In this paper, we collect and summarize the recent research on underwater computer vision with deep learning, focusing on classification, detection, and segmentation of monocular RGB images. To the best of our knowledge, this is the most comprehensive survey paper on deep learning techniques applied to underwater images. Its key contributions are:

- A survey of deep learning techniques for enhancement, classification, segmentation, and object detection in underwater images;
- A survey of the available underwater image datasets;
- A discussion of future work directions, including the benefits of using predictive uncertainty and good practice for developing and training models.

State of the Art: Several surveys address related areas, albeit none of them is comparably comprehensive [26]–[31]. All focus on limited tasks: underwater image classification [29] and underwater object detection [26], [28], [30], [31], not necessarily using deep learning and camera images. Moniruzzaman and coauthors discuss only about ten papers on fish, coral, and plankton classification and detection with deep learning [26]. Teng and Zhao study object tracking [27], including acoustic images, covering only ca. 15 papers on optical images and deep learning. Sarkar et al. survey object detection, in both optical and sonar images using learning-based and non-learning-based approaches [28]. They refer only about ten papers on optical images and deep learning. Wang and colleagues [31] cover both surface and underwater images; only eight papers and two datasets mentioned regard the underwater environment.

The present survey differs from the others by including papers on image enhancement, image classification and segmentation, and object detection. Furthermore, it also includes a list of datasets for performing the above tasks. We discuss 13 papers in image enhancement, 7 in classification, 20 in detection, and 12 in segmentation. Furthermore, we identify three datasets for classification, eight for detection, eight for segmentation, and two for enhancement.

Paper Organization: Section II details how the papers for the survey were selected. Section III discusses the identified datasets. Sections IV to VII are respectively devoted to image enhancement, classification, object detection, and segmentation. We discuss the state of research in underwater vision in Sect. VIII, including good practices for accelerating the work in this field. Finally, Sect. IX concludes.

II. Method to Collect and Select Papers

The search was executed using Google Scholar, with keywords related to the survey's main subject. We used the advanced search to ensure that the query is matched against titles only. We included papers using the following keywords: '*object*,' '*image*,' '*classification*,' '*detection*,' '*recognition*,' '*segmentation*,' and '*underwater*'; more precisely three keywords were required, one from each box in Fig. 3. We excluded papers older than 2012 and those whose title contained '*sonar*,' '*ultrasound*,' '*ultrasonic*,' '*acoustic*' and '*acoustics*' to eliminate research not based on deep learning and not using camera images.

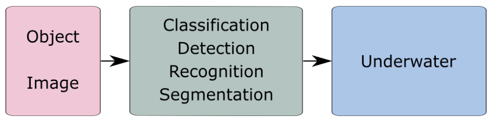


Fig. 3. Keywords in the inclusion query for Google Scholar

We used SerpApi [32], a Google Search API that allows scraping. The process produced Excel tables with the titles of the papers, freezing the list of the papers in scope. The further selection was based on the relevance of each paper to the object of study. We added further papers in a snowballing fashion, when reading the papers identified by the query. Finally, we found the references to many public datasets in the surveyed papers and generated a list with the datasets' names and specifications.

III. Datasets of Underwater Images

We now present the identified datasets. If a dataset has more than one application (labeling), it is discussed in the subsection that best covers both tasks. All datasets are listed in Tbl. I and Fig. 4 shows examples of images from selected datasets.

Datasets for Image Enhancement: The EUVP is a dataset of varying quality underwater images split in two parts [33]. The unpaired dataset EUVP-Unpaired is separated into good and poor-quality images based on the perception of six human judges. The EUVP-Paired part contains images paired with their enhanced version. A limitation is that the poor images result from the CycleGAN model, trained to distort good images, thus they may differ from images captured in the same scene in low visibility moments. Figure 4 shows two pairs of poor and enhanced images from the paired dataset along with two poor and one better-quality sample from the unpaired dataset.

Datasets for Multi-Label Image Classification: MLC is a coral dataset with 2055 images [2], [34], each annotated by human specialists in 200 points, with 9 different classes (5 coral, 4 non-coral classes) giving over 400k annotations. BENTHOZ-2015 is a benthic (the lowest levels of water) dataset with 9874 underwater images of Australia's coast, each annotated with up to 50 points [35]. The images are georeferenced and hierarchically classified into 148 substratum and biological classes.

Datasets for Object Detection: The labeled-fishes-in-the-wild dataset [36] contains images of fish, invertebrates, and seabed taken by a remotely operated underwater vehicle (cf. Fig. 4). A total of 4096 images, labeled with bounding boxes, are provided for validation and training: 929 are positive still images and 3167 are negative samples (no object present). The test set is composed of video frames, which are more challenging to process than still images. It contains 210 frames with 2061 fish objects in total. The training images were captured in the Ocean around Southern California. The OUC-VISION dataset [37] has been produced in a pool simulating

TABLE I
DATASETS FOR IMAGE ENHANCEMENT, CLASSIFICATION, OBJECT DETECTION, AND SEGMENTATION TASKS. SIZE IS GIVEN IN THE NUMBER OF IMAGES

Name (linked)	**Task**	**Size**	**Contents**
EUVP-Unpaired	image enhancement	6665	divided into poor and good quality images, 95% training, 5% validation
EUVP-Paired	image enhancement	13405	divided into poor and good quality images, 85% training, 15% validation
MLC	classification	2055	non-coral classes: crustose coralline algae (CCA), turf algae, macroalgae, and sand; and coral genera classes: *acropora*, *pavona*, *montipora*, *pocillopora*, and *porites*
BENTHOZ-2015	classification	9874	148 substratum and biological classes
labeled-fishes-in-the-wild	detection	4996	fish, invertebrates, and the seabed; 929 positive, 3167 negative samples, 210 video frames of test material
OUC-VISION	detection	4400	salient objects (like rocks). Unfortunately we were unable to locate teh dataset online, despite the authors claiming that it was available. The corresponding author is Muwei Jian (jianmuwei@ouc.edu.cn)
Brackish	detection	14674	fish, small fish, crabs, shrimps, jellyfish, and starfish (80% training, 1468 testing, and 1467 validation)
UDD	detection	2227	sea-cucumbers, sea-urchins, and scallops (1827 training, 400 testing)
DUO	detection	7782	holothurian, echinus, scallop, and starfish (6671 training, 1111 testing)
FathomNet	detection	(growing)	mid-water and benthic objects
TrashCan	detection, segmentation	7212	trash (with object name and material), man-made objects intentionally placed in the scene, bio (animal or plant), and unknown
SUIM	segmentation	1635	background (water), human divers, aquatic plants and sea-grass, wrecks or ruins, robots (AUVs, ROVs, instruments), reefs and invertebrates, fish and vertebrates, sea-floor and rocks; includes 110 test images
DUT-USEG	segmentation, detection	6617	sea cucumber, sea urchin, scallop and starfish (1487 are labelled for segmentation)
NAUTEC UWI	segmentation	700	foreground and background
MarinaPipe	segmentation	1723	pipeline and background, the images are divided into 7 folds, corresponding to frames from 7 videos
SubPipe	SLAM, detection, segmentation	647	pipeline and background, includes side-scan-sonar images for object detection
LIACI	segmentation	1893	defects, corrosion, paint peel, marine growth, sea chest gratings, overboard valves, propeller, anodes, bilge keel, and ship hull
DeepFish	classification, detection, segmentation	620	fish (about 50/20/30% split into training, validation, and test)

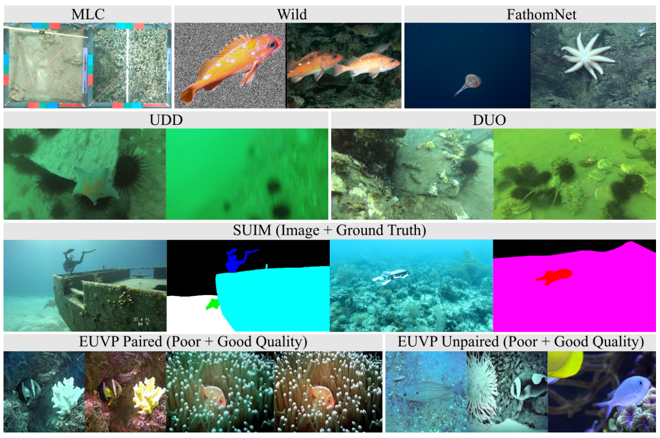

MLC
Wild
FathomNet
UDD
DUO
SUIM (Image + Ground Truth)
EUVP Paired (Poor + Good Quality)
EUVP Unpaired (Poor + Good Quality)

Fig. 4. Examples from the identified datasets. For SUIM (segmentation) we show the ground truth mask as well. For EUVP paired (enhancement) we include both the original good-quality and distorted images. For EUVP unpaired, the first two images are low-quality, and the last one is a better-quality example. MLC dataset is under the CC-BY-4.0 license.

the ocean floor, with varying turbidity and illumination. It comprises 220 salient objects, each photographed in five positions with four different poses per position (4400 images of 2592×1944 pixels cropped to 486×648 pixels to decrease storage demand). Each image contains only one salient object, labelled by the coordinates of its bounding box. The brackish dataset [38] was produced nine meters below the water surface, under varying illumination, in the brackish (somewhat saline) water of Limfjorden, Denmark. It contains 14674 frames from 89 videos with 25613 annotated events such as fish, small fish, crabs, shrimps, jellyfish, and starfish. These are split into training, validation, and testing, and the ground truth is available in several formats (AAU Bounding Box, YOLO Darknet, MS COCO). Multiple events are annotated per frame. The *underwater open-sea farm object detection dataset* (UDD) contains 2227 images from 4K quality videos captured on open sea farm by robots and divers [39]. The dataset is divided between 1827 training and 400 testing images including a total of 1148 sea-cucumbers, 13592 sea-urchins, and 282 scallops. The authors have trained a Generative Adversarial Network (a GAN) and created a large scale augmented dataset (AUDD) that contains 18000 images, addressing the class imbalance. Furthermore, a pretraining dataset was proposed, containing cropped images from other datasets to adapt models to the underwater environment. All three datasets are available online.[1] The same authors created another aggregated dataset DUO [40] gathering URPC2017, URPC2018, URPC2019,

[1]https://github.com/chongweiliu/UDD_Official

$\text{URPC2020}_{\text{ZJ}}$, $\text{URPC2020}_{\text{DL}}$ and UDD datasets, annotating them in COCO format and fixing problems of incomplete and wrong labels, duplicate and similar images, etc. DUO contains 6671 images for training and 1111 for testing featuring 74515 objects, including holothurian, echinus, scallop, and starfish. The dataset is highly unbalanced. See Fig. 4 for examples from UDD and DUO. Finally, FathomNet is a continuously maintained platform, modeled after ImageNet and COCO. It allows to search and select images of both mid-water and benthic objects [41]. By 2020, it contained more than 60k entries. Figure 4 shows two samples from FathomNet.

Datasets for Image Segmentation: TrashCan [42] is a dataset including bounding-boxes and segmentation masks inside them. It was developed to train trash detectors for underwater robots. TrashCan has two versions: 'TrashCan-Material' labeled by material composition and 'TrashCan-Instance' labeled with object categories (can, clothing, pipe, etc.). In both variants, the objects were first classified as trash, man-made item intentionally placed in the scene, bio (animal or plant), and unknown. Images classified as animals are also tagged with their species (e.g. fish, crab). Trash objects are labeled by type (category/material composition), if more than 50 instances of the same class are found in the dataset. Segmentation of underwater imagery (SUIM) [43] is a set of 1635 pixel-level-annotated images with eight classes, in varied resolutions: e.g., 1906×1080, 1280×720, 640×480, and 256×256. Out of these, 110 are earmarked for testing. The images were collected during oceanic exploration and human-robot-interaction experiments. Figure 4 shows two examples from this dataset along with their respective masks. DUT-USEG: DUT Underwater Segmentation Dataset contains 6617 images of real scenes, of which 1487 have semantic and instance segmentation annotations. The remaining images only have the bounding box labels. DUT-USEG has been constructed with the goal to help the development of methods for grabbing objects. The NAUTEC UWI dataset contains 700 wild underwater environment images collected online. They show divers, marine life, and other objects, annotated with masks for background and foreground solely [44]. Interestingly, the authors generated also other datasets *indoor* [45] by simulating back-scattering degradation to obtain underwater visual effects. MarinaPipe [21] and SubPipe [22] are datasets with submarine pipeline images. MarinaPipe contains frames of videos of pipe fragments recorded in the bottom of a marina in Porto and annotated with precise and coarse segmentation masks, cf. Fig. 1. SubPipe, also recorded in Portugal, targets SLAM, object detection and pipeline segmentation tasks. For object detection, SubPipe uses side-scan sonar images, for segmentation it uses RGB camera. LIACI [46] is a dataset recorded on the Norwegian coast that also targets more industrial applications. It consists of images for underwater ship inspections with pixel-level annotated classes, including defects, corrosion, paint peel, marine growth, sea chest gratings, overboard valves, propeller, anodes, bilge keel, and ship hull. Finally, the DeepFish [47] dataset contains about 40 thousand images collected from 20 different Australian habitats. While it has originally featured classification labels only, 3200 images have been enriched with point-level annotations (the coordinates of the centroid of each fish), and 620 images have pixel-level masks.

IV. Enhancement of underwater images

Oceanic water jeopardizes the effectiveness of image processing. Artificial light is typically used underwater to increase visibility in the images while taking them. Unfortunately, artificial illumination suffers from the same problem as natural illumination and, in addition, generates non-uniform bright spots in the images [23]. Therefore, much computer vision research has been devoted to enhancing the quality of underwater images. Although many classical rule-based methods exist, (deep) learning-based approaches are gaining interest.

A. *Simple CNNs Trained with Paired Images*

Li et al. trained a CNN (named UWCNN) with ten convolutional layers to improve contrast and illumination, and to reduce color distortion. A post-processing step on top of UWCNN further improves contrast, and normalizes saturation and intensity in the HSI color space [48]. Other CNNs, with only six convolutional layers, were trained to remove haze [49], [50]. The three works above used pairs of original and enhanced images from real and synthetic datasets for training. The first model used a skip connection, inspired by residual CNNs, which helps to preserve details. Their model was trained against MSE and SSIM losses jointly. The SSIM loss combines the mean, standard deviation, and cross-variance of pixel values, in patches from the ground truth and the reconstructed enhanced images, thereby preserving the structure and texture on the enhanced image. The other two models used the L2 metric.

Chen et al. combine an equation modeling the underwater image formation with two CNNs [51]; one implements a backscatter estimation module, the other a direct-transmission estimation module. The image formation model uses the original image and the outputs of the two CNNs to enhance images. The framework is trained end-to-end using pairs of normal and enhanced images, but instead of learning the transformation between a raw and an enhanced image directly, like the above works, it exploits the image formation, guiding the networks to learn the parameters of light coming directly from the object and from the surrounding. The authors show that the number of matched points using SIFT [52] and RANSAC [53] between a pair of images increases significantly after enhancing both. Figure 5 shows some examples of underwater image enhancement. The enhanced images are clearer, and more visually appealing, without the green tone caused by the color deviation.

B. *Improving Illumination with CNNs and Retinex*

Retinex, a theory of color vision and objects reflectance [54], postulates that an image can be decomposed into reflectance and illumination maps [55] that can be used to enhance the illumination quality. Han and coauthors trained a 4-layer CNN to model the relation between the images and the respective maps, and used them to enhance illumination with the retinex model [56]. Although the example images presented in the article are brighter, the light gets too strong at some areas of the enhanced images, making objects less visible, cf. Fig. 5. A method named LigED was developed to improve lightning and eliminate darkness in images of live crabs [57]. LigED uses a

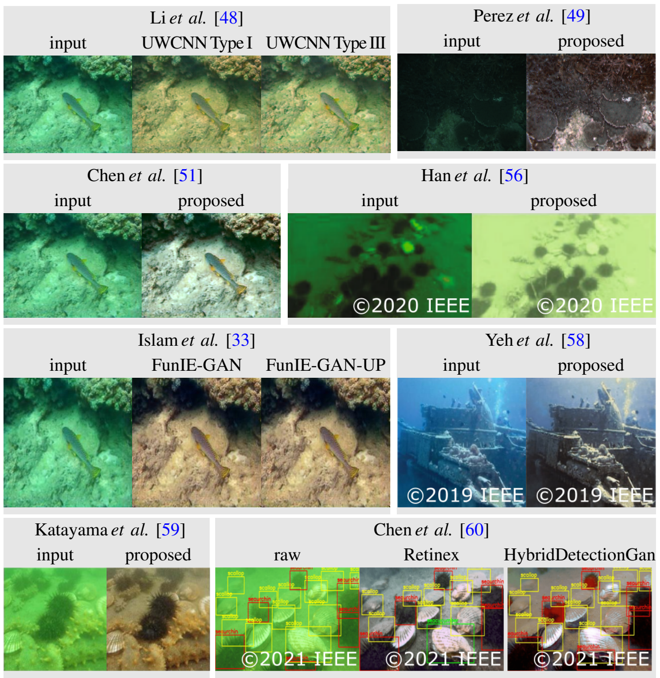


Fig. 5. Underwater image enhancement. 'Input' are the original images, 'raw' is the input with ground truth. The images were reproduced by us with publicly available code or models, or found directly in publicly available datasets. The work of Han *et al.* [56] is licensed under CC-BY-4.0.

camera response function model to generate strong-light images, retinex theory to create illumination and reflectance maps, and convolutional-based neural networks. LigED generates more maps than Han and colleagues. For evaluation, they train an object detection model, showing that the average precision of the model increased from 81% to 95% after the preprocessing.

C. GAN-Based Structures

Paired lo–hi images are difficult to obtain. FUnIE-GAN [33] circumvents this by using a generator inspired by U-net [61] and training on both paired and unpaired images from the EUVP dataset. The objective function evaluates color, texture, style, and global content to ensure a better visual quality of the image. A cycle consistency loss is used for the unpaired training. The neural network is trained to perform transfer style, to imitate human perception of quality. Figure 5 shows an example obtained with FUnIE-GAN. Fabbri et al. trained a CycleGAN with a manually selected unpaired dataset of underwater images with and without distortion [62]. Then they used the trained model to generate a paired dataset to train UGAN, a model for enhancing images. Yeh and colleagues propose a framework based on three simple CNNs trained on underwater and in-air images [58]. The tasks of the CNNs are (1) to convert the images from RGB to greyscale, (2) to enhance the greyscale images, and (3) to restore the color image quality. The outputs are combined based on hue preservation, and a CycleGAN architecture is used for training; see Fig. 5 for an example. This allows to exploit the more easily available high quality in-air images.

Noting that detection models perform better on in-air images than in water, Katayama et al. trained a CycleGAN with a polynomial loss function, including the structural similarity index measure loss, to transfer style from underwater to in-air [59]. Curiously, YOLO [63] trained with the generated in-air images performed better on the original underwater images than on the same images with style transferred to in-air. The images used to train the GAN-based style transfer model were taken from a subset of underwater and in-air images in ImageNet. YOLO was trained and test with a subset of ImageNet containing sea cucumber, sea urchin, and scallop. See also Fig. 5.

D. Enhancing Images for Object Detection

Enhancing images towards the human notion of quality is not necessarily the best choice to improve the performance of computer vision. Considering this, Liu and coauthors [64] trained a GAN-based model to generate enhanced images of divers, to specifically benefit detection models, not human perception. The objective function of the GAN was modified to include information from a SSD model [65] with a MobileNetV2 backbone. They tested different setups, including this information in different parts of the GAN: the generator (DUnIE-GAN-G), the discriminator (DUnIE-GAN-D), and both (DUnIE-GAN-B).

HybridDetectionGAN is another method aiming to improve detection tasks [60]. HybridDetectionGAN uses a CycleGan [66] that learns style transfer between in-air and water images simultaneously with learning object detection. Physical priors are used to improve the realism of the generated images. Figure 5 shows that detection is more accurate in the image enhanced with HybridDetectionGan than with Retinex [67].

The Underwater joint image Module (UnitModule) learns the parameters of an equation that explains the underwater images formation [68]. It can be plugged into a detection model, enhancing images in a way that improves object detection. The authors used a color-casting predictor and a data augmentation technique, the underwater color random transfer, that considers the average hue of underwater images.

E. Discussion

Zhang et al. [69] analyze the relationship between the performance of image enhancement and object detection. They compared (1) a classic enhancement method comprizing dehazing, color compensation, histogram equalization, and a bilateral filter, (2) UWCNN, and (3) FUNIE-GAN. Although enhancing images improves object detection, they assess that the minor improvements observed are not enough to justify an image enhancement before deep-learning-based object detection. This is contrast with the works reported above that consistently show that the computer vision perception of the images can be notably improved. Techniques used range from, simple CNNs with supervised learning, frameworks that use physic theories, and GANs [70], depending on the available data. Supervised training methods require ground truth or ways to generate it (image pairs, illumination maps, or camera response function). These structures are easier to model, but it is harder to prepare the data for them. In contrast, GANs are more complex than simple CNNs, but do not require high-low quality image pairs. Furthermore, as demonstrated [60], [64] it is possible to incorporate the task performance metrics into the training process for image enhancement models.

V. Classification

Classification means assigning a class to each image based on its content, e.g. a diver or a cracked pipeline. Many powerful CNNs structures were developed [71]–[75], not least thanks to the ImageNet competition. These results encouraged many researchers to test whether these structures would perform well for other datasets. Unfortunately, training a deep learning model requires vast data, while data collection and annotation are expensive, especially so in the submarine environment. As training from scratch with limited data usually does not lead to good results, many use transfer learning and fine-tuning, taking advantage of pre-trained CNNs for other domains.

A. Feature Extraction

The features extracted by CNNs are expressive in the sense that they can usually be transferred from one task to another. Cao and coauthors [76] trained a SVM [77] using a combination of handcrafted features, and features extracted from the fully connected layers of a pre-trained AlexNet [71]. More CNN than handcrafted features were used, but the latter helped with the robustness when the quality of the images was decreased. The structure was trained with two datasets, one with fish and the other with benthic animals.

To take advantage of the powerful capability of pre-trained CNNs of extracting semantic information, the image features ResFeats were developed using feature maps from different convolutional layers [78]. The ResFeats were used to help in the classification of benthic datasets, including MLC [2] and BENTHOZ-2015 [35]. The approach consists of extracting the feature maps from the output of different residual blocks of a ResNet-50 [73] pre-trained in ImageNet, and max pooling these maps to obtain 1D vectors with the same size as the number of channels in the output of the respective residual blocks. The concatenation of the resulting feature vectors was used to train an SVM [77] classifier.

Going in the opposite direction, after applying many classic image pre-processing techniques, feature vectors for shape matching, texture, and color description were obtained and used as input for different machine learning algorithms, such as SVMs, K-NNs, random forests, CNNs, and deep fully connected neural networks [79]. Although it is not usual to extract features to use as input to deep learning approaches, since these models should perform the feature extraction, the authors showed that better results were obtained by the deep models than by the classic machine learning classifiers.

B. Fine Tuning in Small Size Datasets

Towards the direction of end-to-end approaches and still dealing with tiny datasets, fine-tuning was used for classifying fish species [80]. An improved median filter was applied to the pixels corrupted by noise, which helps to preserve the details in the images, and a CNN similar to AlexNet [71] was trained first in the ImageNet dataset, and then fine-tuned with a small dataset of fish species with 500 images for training and 200 for validation. An interesting detail in this research is that, during the fine-tuning phase, they used a learning rate ten times bigger for the weights in the last layer of the CNN. In another research, the performance obtained with transfer learning was observed for several models [81]. This study used a dataset with 600 images, divided between 70% for training and 30% for validation. The images belong to the classes divers, unmanned underwater vehicles, fish, and water. Data augmentation was applied, resulting in 2400 images. Various pre-trained models, such as AlexNet [82], Inception-v3 [83], VGG-16 [72], DenseNet-201 [75], and ResNet-50 [73] were tested. The authors used backpropagation with SGDM, RMSProp, and Adam [84] for transfer learning, and a genetic algorithm was used to decide what hyperparameters to use for training. Using AlexNet as a comparison, Adam obtained the best results regarding average accuracy and training time. After choosing the Adam algorithm, and applying it to all the networks, the highest accuracy in the validation dataset was obtained by DenseNet-201.

C. Adversarial Learning

Classification tasks can benefit from the power of adversarial learning. A model with low and high features alignment was used in combination with adversarial learning to perform the domain shifting between an in-air dataset with man-made objects with labels and a smaller dataset of the same objects in the underwater [85]. Labeled and unlabelled images from the underwater dataset were used for training, and encouraging results were obtained even when using only one labeled underwater image per class.

Nonetheless, the previous authors used the same classes both for in-air and underwater, which can be not possible sometimes. Using a strategy that does not require matching classes in two domains, GAN was applied to triple the underwater images from imageNet, increasing the accuracy of a GoogLeNet [74] (a particular Inception structure) trained with this subset [86]. The authors also tested to double this dataset with conventional data augmentation. Compared with the approach mentioned in the last paragraph [85], the downside of only augmenting the dataset is the necessity for more labeled data from the target domain.

D. Image Classification Discussion

The subsections above introduced approaches that were used by different researchers working in the underwater environment. They took advantage of well-established models and techniques and adapted them to obtain better results in their study cases.

VI. Detection

Typically, in deep learning, if a model performs well in a task, it will also works well in a similar problem. This principle is frequently used in image processing. Many of the CNNs that achieved state-of-the-art are available online with the pre-trained weights, and many researchers and companies use these pre-trained models for their particular tasks. As explained earlier in the preceding section, transfer learning and fine-tuning are methods commonly used to take trained models and apply them to new tasks without massive datasets or much computational effort. It is possible to use classic models, usually pre-trained in datasets such as ImageNet, for underwater problems, e.g., fish detection and recognition [87]–[89] with YOLOv3 [90].

Nevertheless, the images of the fauna and flora of underwater environments typically have small, occluded, and overlapped objects. The most well-known pre-trained deep learning based object detection models, such as R-CNN [91], YOLO [63], [90], [92] and SSD [65], have difficulty dealing with objects with this disposition in images. This limitation can be due to the neural network structure itself and the difference in the data used to pre-train the models.

The available pre-trained models have been trained in datasets such as imageNet or COCO, which contain mainly in-air images. These images do not have the same issues as underwater images, pointed out before, generating the problem of different data patterns. The use of enough underwater images to fine-tune the models is a way to overcome this problem. Another issue concerning the datasets used for pre-training is that the images contained in them usually do not have as many overlapped objects as underwater images. This last problem is probably more challenging to solve. However, fine-tuning with enough data can also fix it.

When it comes to the structure of the models, many researchers are concerned with the current models' ability to deal with the detection of small objects. The filters in the deeper layer of a CNN have a larger field of view [93], resulting in the extraction of more complex patterns [94]. In comparison, the first layers of the model recognize colors and basic patterns [94]. This way, deeper models are more potent in extracting semantic information, resulting in better classifiers. Nevertheless, the size reduction in the image's dimensions, that occurs while the model gets deeper, can imply a loss of information about small objects. That way, it jeopardizes the model's capability to identify small objects and, more specifically, to detect their position in the image. Between the computer vision research for the underwater environment, much effort is put into solving the small objects detection problem.

The following subsections aim to discuss the different strategies used by researchers to test the CNN models or modify typically used structures. It is hard to classify the changes implemented by the authors since many structures use many modifications at the same time, and these changes in the structure were not continually developed in a chronological or structured way, meaning that one model would aggregate all the previous structure updates before receiving a new one. Notwithstanding, this section classifies the papers trying to observe the potential effects of introducing new strategies in the models.

A. Transfer Learning and Fine-Tuning

To compare the effectiveness of changing the structure of the state-of-the-art CNNs or the necessity of creating new models from scratch, it is interesting to first observe how transfer learning and fine-tuning works for underwater images. It is also interesting to test small changes in the models that do not add layers, or that decrease the size of the CNN, favoring the processing speed.

A research was developed to compare different strategies for training YOLOv3 [90] to detect and classify two species of fish [1]. First, the model was trained without transfer learning using 200 images. Then, the authors added 200 more images to the dataset and retrained the model, increasing the mAP for training from 53% to 74%. Finally, it was used a YOLOv3 pre-trained with ImageNet, improving the mAP by 4%.

Instead of straightly transferring the learning from a pre-trained model to the desired dataset, it was tested a two-steps strategy to slowly transition between different image patterns [95]. First, using the weights pre-trained with imageNet, a Faster R-CNN [96] was trained with a dataset to detect fish, with images with resolution about 1920x1080 pixels and high contrast e sharpness. Then the previous CNN was transferring learned again to a fish dataset of images of smaller size and with less contrast and more blur. In this last step, only the deeper layers were retrained. The authors also tested to enhance the low-resolution image dataset with Retinex before the last transfer learning step. Finally, the authors observed that decreasing the training learning rate leads to better mAP.

Very similarly to the previous strategy, CMA was introduced as a concept inspired by the human way of learning easier examples first [97]. It is a continuation of the work presented in SWIPENET[98], which will be introduced in a further subsection. With CMA, the CNN first learns to detect objects in a 'clean' detector, that uses sample weights that discourage the learning of objects that the model does not detect. This strategy considers that these undetected objects are probably in noisy images and harm the learning of the model. After that, many detectors are trained using as a base the 'clean' detector. In this second phase, the training is done in an inverted way, giving the samples with undetected objects more weight. Finally, the resulting detectors were combined in an ensemble.

The authors from [99] used the structure of SSD [65] and transfer learning, but slightly changed the model to deal with the problem of small objects. This change was introduced by adding skip connections, without introducing new layers. Since the shallower layers have more detailed spatial information than deeper layers, the conv2 and conv3 layers from the backbone of the original SSD model were connected to the detection part of the structure. The dataset used in this paper was provided by 2018 UROGC. Since the training of this model also used transfer learning, beyond dealing with small objects, this approach is more accessible to be used when only small datasets are available, considering that pre-trained weights are publicly available.

A next step, related to transfer learning and highly important to allow models to work in practical applications, is to test the ability of the detectors to generalize. In the real world, it is impractical and often impossible to train a model with data from the same spot where it will be utilized. Therefore, it is desirable that a model trained with a dataset recorded in one site performs well with data from another place. As a minimum, the model needs to work for a dataset recorded at another moment in time, for example, in another AUV survey. Simulating this situation, a YOLOv3 [90] model was trained for detecting fish using two real-world datasets and tested in a third dataset [100]. The three datasets were recorded at different spots. As a result, it was found that the model could not recognize fish in the test dataset. In other experiments, it was observed that training the model with only two datasets resulted in a better performance in the images of these two datasets than training with all three datasets. These results show the challenge in developing models able to generalize and the importance of paying attention to the overfitting while designing CNNs. Moreover, it highlights the importance of carefully using a training dataset with the same pattern as the images in which the model is supposed to work in real life.

Another step regarding the application to real-world problems is to test reducing the size of pre-trained models to reduce the processing time during inference. To speed up the model, it was tested to shrink a pre-trained MobileNetV2 used as the backbone of a YOLOv2 [63] by removing blocks of the structure [101]. The model was trained with a dataset of 5833 goldfish images of six breeds. Starting with a 16-block MobileNetV2, the model's performance was tested after successively removing the blocks. When comparing the 16-block structure with the 1-block structure, the number of trainable parameters was reduced from more than 3 million to 17,328, increasing the detection velocity from approximately 12 to 56 fps. However, it only decreased the mAP from 95% to 89%.

Table II shows some of the results obtained by the researchers mentioned in this subsection. From this table, it is possible to see the improvement in results when implementing transfer learning or transfer learning-inspired strategies.

B. Features Mixing with Standard Models

Numerous attempts to improve objects detection have been made in the last few years. Many of them try to modify well-known models inserting new layers, creating new ways of applying convolution, or trying different skip connections or mixing features from different convolutional layer levels. However, a promising direction of research could be using the standard CNN models and mixing their structures to take advantage of the good results already obtained by that models.

FERNet [102] mixes the outputs from different layers of VGG-16 [72] and ResNet-50 [73] to generate the backbone called CCB. The authors used this strategy to improve the features extracted, aiming to help with blur and distortion problems. After the backbone, to increase the capability of detecting objects of different scales, RFAMs were applied. These modules are a reproduction of the work developed by the authors of RFBNet [103], and uses strategies from Inception [74] and ResNext [104], such as shortcuts, convolutional kernels with different filter sizes, dilation factors, and stride values. Finally, the results extracted by the RFAMs are processed by a module that the authors call PRS. This module classifies the objects between background and foreground, then applies a deformable convolution network, and finalizes with a multi-classes classification. The PRS was used to help align the features extracted with the anchor boxes and to help with the class imbalance problem. For testing the CNN developed, a dataset with 10 thousand images and four classes (holothurian, echinus, scallop, and starfish) was used.

Still using feature mixing, a strategy that can be interpreted as data augmentation was used to simulate overlapped objects during training [105]. A RoIMix module was inserted between the RPN and the Classifier of a Faster R-CNN [96]. The RoIMix module consists of a block that takes two random RoIs from multiple images and adds them, keeping the label of the RoI that contributes more to the mix. Using the dataset URPC 2018, the authors improved the mAP of the Faster R-CNN baseline by 1.18%.

Table III analyses the results of using the feature mixing strategies explained in this subsection. For FerNet comparison, the authors used RFBNet [103]. It is important to observe that the CCB backbone is not the only difference between FERNet and RFBNet structures, and that the result obtained by FERNet was achieved using improved strategies for training. However, to specifically analyze the use of feature mixing, it can be seen in table III that RFBNet with CCB as the backbone resulted in an improvement of the mAP by almost 4% when comparing RFBNet with VGG16 as the backbone.

Finally, it could be interesting to test the use of both strategies with other well-known models. For example, FERNet [102] could be developed using other CNNs, such as AlexNet or DenseNet, and the ROIMix module [105] could be implemented with other 2-stages detectors.

C. Layers Resolution Increase

As explained before, the image size reduction that occurs as it passes through the CNN can result in the loss of small objects' information. A strategy to detect objects with different scales, including small objects, is to increase the resolution of the feature maps extracted by the CNNs at the deeper layers and use feature map combinations from different layer levels in the detector block of the model. Further, to keep track of detailed information about the position of the objects, especially the minors, skip connections from the shallower layers can be used. Figure 6 shows examples of resolution increase in CNNs.

The same paper that released the dataset UDD [39] proposed the CNN AquaNet, which uses the strategy of increasing the size of feature maps to better detect small objects. They implemented MBP blocks, which use normalized Gaussian filters with three different smoothings, to deal with the blur problem. For solving the problem with small objects, they applied MFF blocks, which use depth-wise separable convolution [106] with different kernel sizes and 1x1 standard convolutions. After a sequence of MBP and MFF blocks, bilinear upsampling was used to increase the dimension of the feature maps and add them to the shallower maps. Finally, the resulting feature maps were used to detect the objects.

M-ResNet [107] also uses the approach of increasing the resolution of the feature maps, but instead of generating one final set of maps as in AquaNet [39], it utilizes maps from different levels to detect objects in different scales. M-ResNet uses a modified ResNet [73] and generates three combinations of feature maps, using the outputs of different convolutional layers merged with the help of upsampling, to detect small, medium, and large objects. In addition to upsampling layers, dilated convolutional layers with different rates increased the

TABLE II
COMPARISON OF DETECTION MODELS FOCUSED ON TRANSFER LEARNING. THE MAP RESULTS ARE TAKEN FROM THE REFERENCED PAPERS

Model	**Backbone**	**Dataset**	**mAP**	**Comments**
SWIPENET-noCMA [97]	SWIPENET	URPC2018	0.622	Model trained without the CMA strategy
SWIPENET-Single [97]	SWIPENET	URPC2018	0.653	The best single (without ensemble) achieved with CMA
SSD [99]	VGG16	2018 UROGC	0.609	TL=Yes
Improved SSD [99]	VGG16	2018 UROGC	0.656	TL=Yes
YOLOv3 [1]	—	2 fish species	0.53	200 images for training without transfer Learning
YOLOv3 [1]	—	2 fish species	0.74	400 images for training without transfer Learning
YOLOv3 [1]	—	2 fish species	0.78	400 images for training with transfer Learning
Faster R-CNN [95]	VGG-16	fish detection	0.791	TL=No
Faster R-CNN [95]	VGG-16	fish detection	0.803	TL=Yes
Faster R-CNN [95]	VGG-16	fish detection	0.898	TL=Yes Images enhanced by Retinex
Faster R-CNN [95]	ZFNet	fish detection	0.708	TL=No
Faster R-CNN [95]	ZFNet	fish detection	0.764	TL=Yes
Faster R-CNN [95]	ZFNet	fish detection	0.791	TL=Yes Images enhanced by Retinex
YOLOv3 [100]	—	Datasets (1), (2) and (3)	Dataset (1) = 0.5474 Dataset (2) = 0.5575 Dataset (3) = 0.4507	TL=Yes 3 datasets for fish detection were used for training: (1) Voith Hydro; (2) Wells Dam; (3) Igiugig
YOLOv3 [100]	—	Datasets (1) and (2)	Dataset (1) = 0.5714 Dataset (2) = 0.5659 Dataset (3) = 0.0055	TL=Yes 2 datasets were used for training

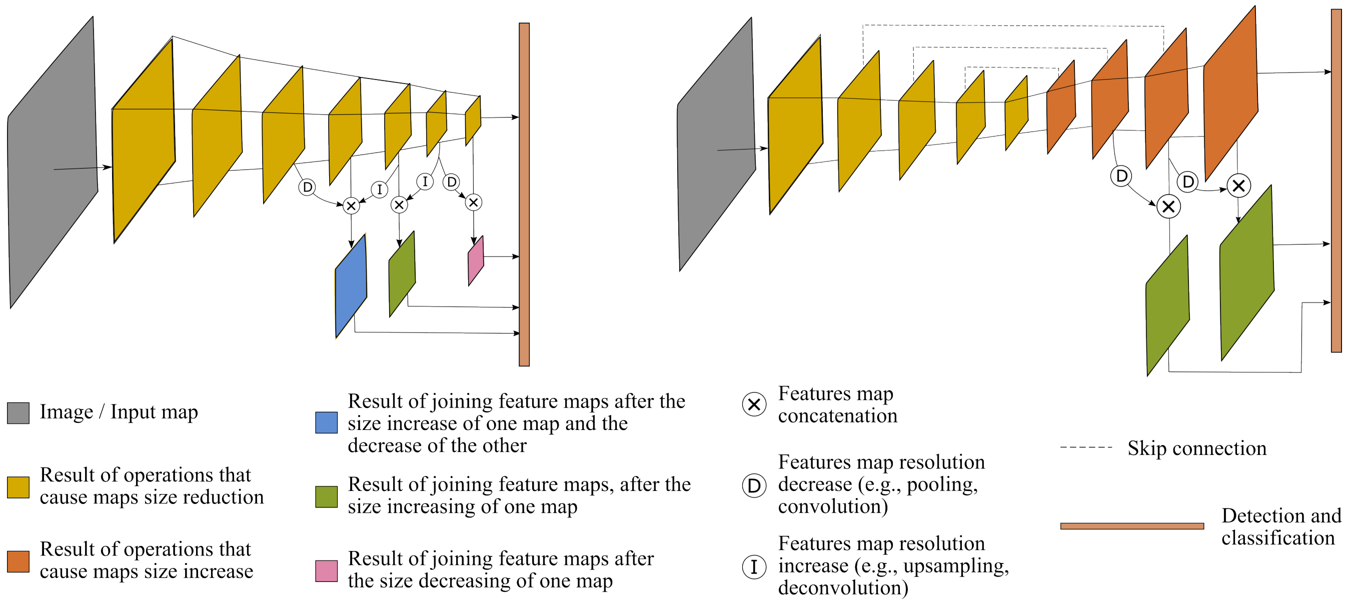

D
I
Image / Input map
Result of operations that cause maps size reduction
Result of operations that cause maps size increase
Result of joining feature maps after the size increase of one map and the decrease of the other
Result of joining feature maps, after the size increasing of one map
Result of joining feature maps after the size decreasing of one map
Features map concatenation
Features map resolution decrease (e.g., pooling, convolution)
Features map resolution increase (e.g., upsampling, deconvolution)
Skip connection
Detection and classification

Fig. 6. Feature maps combination and increase resolution. The neural network on the left shows different ways of combining feature maps, increasing or decreasing the maps' resolution when necessary. The network on the right shows a resolution increase of the feature maps in 'the main flow' of the CNN. Skip connections are also shown in these images. The models in this figure are a representation to explain the concepts, and do not represent a specific model.

receptive field and captured features from objects of different sizes. The datasets used to test this structure were Fish4 Knowledge Ground-Truth and a dataset collected by the authors. Both datasets are for fish detection.

Another CNN, SWIPENET [98], uses deconvolutional blocks on top of a truncated VGG16 [72] to solve the problem of coarse resolution to detect small objects. Dilated blocks were placed between the output of VGG16 and the first deconvolutional layer to increase the ability to capture semantic information without harming spatial information. Shot cuts were added linking the VGG16 and the deconvolutional blocks to keep track of the spatial details. The skip connections and the resolution increase remember the structure of AquaNet [39], but instead of bilinear upsampling blocks, it was used deconvolution. Instead of only using the last feature maps to input in the detector, as in AquaNet, SWIPENET uses feature maps from different levels of layers, as in M-ResNet [107]. It is worth mentioning that, since SWIPENET uses VGG16, it is possible to use pre-trained weights in this part of the model, helping the training with smaller datasets.

Using the idea that performing the feature maps mixing with layers from different levels before the detection block can improve object detection, a model was developed that performs feature mixing between maps from neighboring layers before

TABLE III
COMPARISON BETWEEN MODELS FEATURES MIXING WITH AND WITHOUT STANDARD MODELS*

Model	Backbone	Dataset	mAP
FERNet [102]	CCB	UnderWater**	0.742
RFBNet [102]	VGG16	UnderWater**	0.60
RFBNet [102]	CCB	UnderWater**	0.638
Proposed (RoIMix) [105]	ResNet-101	URPC 2018	0.7492
Faster-RCNN [105]	ResNet-101	URPC 2018	0.7374

*mAP results taken from the referenced papers
**UnderWater dataset classes = holothurian, echinus, scallop, and starfish

the detection block [108]. The model developed has a very similar structure as SWIPENET [98]. VGG16 [72] was also used as the model's base, but dilated layers were not applied. They studied various strategies for concatenating the maps from different layers, including using 1x1 convolution to half the number of channels, L2 normalization before mixing the features, element-wise sum, element-wise max, element-wise product and channel-level concatenation. The dataset used in this research came from the auto-grabbing contest of 2017 Underwater Robot Picking Contest.

Table IV shows the results obtained by the models mentioned above, together with a comparison with some well-known models. As can be seen, YOLOv3 is only slightly worse than AquaNet when training from scratch and is better when using a pre-training dataset. However, AquaNet has 1.30 million parameters, while YOLOv3 has 61.9 million [39]. It is interesting to notice that the other models, which had a more significant performance improvement compared to standard models, used as the input for the detector block feature maps from different layer levels, while AquaNet only used the feature maps from the final layer of the CNN.

D. Transfer Style and Domain Generalization

As discussed in the subsection about transfer learning and fine-tuning, Sect. VI-A, an essential goal in object detection is to train models able to generalize. Part of the generalization goal is domain invariant models. These models should recognize the semantic information without being harmed by the background change in the image.

Intending to achieve an invariant domain model, the authors from [109] developed a detector called DG-YOLO, which is based in YOLOv3 [90]. First, to obtain a training and validation dataset, a water quality transfer method was used to generate a synthetic dataset simulating eight different water qualities for the same semantic information. Then, YOLOv3 was modified resulting in DG-YOLO, which is a model that detects the objects in the image and classifies the domain. Adding to the model an output that classifies the image domain and a GRL, the YOLOv3 backbone is fooled and forced to ignore the domain information, focusing on the semantic information. For training DG-YOLO, the Invariant Risk Minimization penalty was added to the YOLO loss to help the CNN to learn across multiple domains. Table V makes a comparison between DG-YOLO and YOLOv3 in UPRC2019 dataset. The results presented are for the validation dataset, which is different from the training dataset both regarding domain and semantic information. Using synthetic images with seven different water qualities to train, DG-YOLO performed better than YOLOv3 in the eighth domain dataset, but worse in the original dataset. The authors explain this result by claiming that DG-YOLO is focusing on the semantic information, while YOLOv3, which is not encouraged to ignore the domain information, is using spurious correlations, such as the color that objects have in blueish or greenish images. It is also interesting to notice that training the basic YOLOv3 with the seven synthetic domains leads to better results than training with the original dataset. It could be both because much more data is being used to train the model or because the model is learning to be domain invariant.

A CycleGAN was used to correct the class imbalance, a problem that jeopardizes detectors' training, of the URPC2018 datasets [110]. For that, the first step was to select the images in which most objects belong to the minority classes and few objects to the majority classes. The second step was to divide the datasets URPC2017 and URPC2018 into four categories according to their color distortion: green, blue, deep blue, and white. Then, CycleGANs were trained to perform style transfer from each color distortion to others. Finally, the trained CycleGANs were applied to the selected images with objects of minority classes, augmenting the number of images with these objects. Although the authors demonstrated that the class imbalance problem was solved, they did not present a study showing that this strategy can improve the training of detectors.

E. Attention Module

Since attention modules help the CNN to find the important parts to be analyzed in the images, it can help in the detection of small objects. For the same reason, researchers hope that it improves the model's performance in blur images.

MFFSSD is a CNN that modifies the SSD [65] model by applying spatial and channel attention modules to different outputs of the convolutional layers [111]. The resulting feature maps are mixed in different combinations, and used as input to the detection block together with feature maps from the deeper layers of the developed model.

Another model, instead of applying attention modules in many feature maps from different convolutional layers, applied a single mixed attention block, composed of a channel and a spatial attention modules [112]. This attention block was placed on top of the ResNet-101 used as the model's backbone, and the output of this block was passed through some convolutional layers, and finally, the objects were detected in the resulting feature map.

Instead of using the typical channel attention modules [113] that results in a feature vector, a Coordinate Attention (CA) [114] module was used for helping in the detection of crabs [115]. Instead of a 2D global pooling, this module applies to an input feature map of size (C,H,W) two global average poolings, one with a (H,1) size kernel and the other with a (1,W) kernel, for capturing dependencies along directions and position. Low- and high-layer feature maps were mixed using a feature fusion module (FFM) to capture rich semantic information while keeping detailed location information. The CNN model used was a modified CenterNet with MobileNetv2 backbone.

TABLE IV
MODELS USING LAYERS RESOLUTION INCREASE

Model	Backbone	Dataset	mAP
AquaNet [39] - trained from scratch	-	UDD	0.474
AquaNet [39] - pre-trained with the pre-training dataset from [39]	-	UDD	0.553
YOLOv3 [39] - trained from scratch	darknet53	UDD	0.468
YOLOv3 [39] - pre-trained with the pre-training dataset from [39]	darknet53	UDD	0.578
M-ResNet-152 [107]	ResNet-152	Fish 4 Knowledge	0.923
M-ResNet-101 [107]	ResNet-101	Fish 4 Knowledge	0.903
ResNet-101 [107]	ResNet-101	Fish 4 Knowledge	0.861
YOLOv3 [107]	darknet53	Fish 4 Knowledge	0.887
SWIPENET-BestSingle [98] - The best single model trained	SWIPENET	URPC2018	0.633
SWIPENET-Ensemble [98] - The ensemble of models trained	SWIPENET	URPC2018	0.645
YOLOv3 [98]	darknet53	URPC2018	0.577
Proposed by [108]	VGG16	URPC2017	0.639
YOLOv2 [108]	darknet19	URPC2017	0.487
Faster-RCNN [108]	VGG16	URPC2017	0.567

*mAP results taken from the referenced papers

TABLE V
DOMAIN GENERALIZATION [109]

Model	Dataset	mAP
DG-YOLO	1-7 synthetic domains derived from UPRC2019	Original domain = 0.5481* 8th domain = 0.3377*
YOLOv3	1-7 synthetic domains derived from UPRC2019	Original domain = 0.5856* 8th domain = 0.3055*
YOLOv3	UPRC2019 - original data	Original domain = 0.5645* 8th domain = 0.1637*

*Results taken from the referenced papers

For jellyfish classification and detection [116], YOLOv4-tiny, a model based on YOLOv4 [117], was improved by placing a hybrid attention module after the backbone. The training was performed using mosaic enhancement, which consists of stitching random images together; label smoothing, which considers that the ground truths could have potential errors; and cosine annealing to decrease the learning rate.

Table VI compares the results of the models referenced in this subsection with the respective baselines. One version of the Faster-RCNN improved with a mixed attention module included an image enhancement step [112]; however, the results did not improve significantly. Compared to the respective baseline models, MFFSSD [111] obtained a more significant improvement in the mAP than the model that used a single mixed attention block [112]. This superior performance could be both because MFFSSD used more attention modules and because of the application of feature maps mixing, which helps detect objects with different scales.

F. Deformable Convolutional Layer

Since the deformable convolutional layers have a receptive field area that changes according to the input, it can be an excellent strategy to deal with objects of different sizes.

An architecture using ResNet-101 to extract features was developed to detect sea cucumbers, sea urchin, scallop, and starfish [118]. In this adapted structure, part of the convolutional layers was constructed as deformable convolutional layers. A deformable PS-ROI pooling was used for the neural network's detection block. The structure developed by the authors is composed of a preprocessing module, a feature extraction module based on deformable convolution, and the classification and detection block. The results in Tbl. VII show that the Faster R-CNN using deformable convolutional layers and PS-ROI achieved a better performance than the original Faster R-CNN [96]. The mAP achieved by the model proposed by the authors, which also includes a pre-processing block, is even higher.

G. Training with Incomplete Labeled Datasets

Labeling images is usually time-consuming, which results in datasets missing the annotation of several objects. Furthermore, as previously discussed, there is a particular difficulty in finding datasets for the underwater environment. A two-step model was developed to deal with the problem of incompletely labeled datasets [119]. First, a weakly-fitted segmentation CNN was used to divide the images between regions of interest and background. Then, the segmented images were used to help in the selection of positive and negative detections and eliminate the influence of unlabeled objects in the dataset. The improvement in the model comes from eliminating false negatives due to incomplete annotation from the training and validation datasets. The data used to develop the model comes from the Underwater Robot Picking Contest 2017 dataset (URPC2017). As explained in the article, the method proposed can be implemented using several two-stage models. Using as reference the Faster R-CNN [96], Table VIII shows that the method developed increased the mAP in the URPC2017 dataset by more than 10%.

H. Image Detection Discussion

Often, several changes are implemented in a model at once, making it difficult to understand which modification impacted the results the most. Furthermore, different datasets were used to test the models developed in each article, harming the comparisons between research. Nonetheless, it was demonstrated that, in a general, if pre-trained weights are available, initializing models with them is a good strategy. Increasing the resolution, using feature maps from different layer levels, and skip connections between shallower and deeper layers also result in a good performance. Attention models

TABLE VI
MODELS WITH ATTENTION MODULE*

Model	Dataset	mAP	Comments
SSD [111]	URPC Dataset	0.659	3308 images for training
MFFSSD [111]	URPC Dataset	0.728	3308 images for training
Faster-RCNN [112]	URPC Dataset	0.455	2901 images for training - 4 classes
Faster-RCNN + Mixed attention [112]	URPC Dataset	0.462	2901 images for training - 4 classes
Faster-RCNN + Mixed attention + Enhancement [112]	URPC Dataset	0.467	2901 images for training - 4 classes
YOLOv4-tiny [116]	generated by the authors	92.46	11,926 images - disruptor fish + 7 jellyfish species
Improved YOLOv4-tiny structure [116]	generated by the authors	94.05	YOLOv4-tiny + attention
Improved YOLOv4-tiny algorithm [116]	generated by the authors	95.01	Improved YOLOv4-tiny + mosaic enhanc. + cosine annealing + label smoothing
CenterNet [115]	generated by the authors	95.33**	-
CenterNet+CA [115]	generated by the authors	96.20**	Uses Coordinate Attention
CenterNet+CA+FFM [115]	generated by the authors	97.86**	Uses Coordinate Attention and Feature Fusion

*Results taken from the referenced papers
**Results reported as AP

TABLE VII
MODELS USING DEFORMABLE CONVOLUTION [118]*

Research	Backbone	Dataset	mAP
Regular Faster-RCNN	ResNet-101	URPC2018	0.580
Deformable Faster-RCNN	ResNet-101	URPC2018	0.675
Proposed by [118]	ResNet-101	URPC2018	0.903

*Results taken from the referenced papers

TABLE VIII
MODELS DEALING WITH INCOMPLETE DATASETS [119]*

Model	Backbone	Dataset	mAP
Faster R-CNN	VGG16	URPC2017	0.576
Proposed by [119]	VGG16	URPC2017	0.697

*Results taken from the referenced papers

and the deformable convolution seems to have a positive impact on the models, even though fell work focused on these strategies for the underwater environment. Furthermore, domain generalization is a vital study area that needs more research for underwater images. For the next steps towards developing the field of underwater object detection, it would be great to have a dataset to benchmark, as it is done with imageNet, COCO, or Pascal VOC. Finally, it should be established a structured way of analyzing the impact of each change in the CNNs.

VII. SEGMENTATION

A wide variety of deep learning structures for image segmentation exist and present high performance. However, for the underwater environment, the amount of data available for training is not as vast as for above water, which prevents the field from evolving quickly. The following subsections present some research on underwater image segmentation.

A. Application of well-known models

Despite the challenges in underwater datasets, some researchers successfully applied well-known CNNs for this environment.

The U-net [61] structure was used to segment images of five different fish species between foreground and background [120]. The authors tested to vary the threshold for identifying a fish after the sigmoid output. As a result, the best IoUs were obtained by a threshold between 0.5 and 0.6 for all fish species. Another classic deep learning segmentation model was used to segment underwater images containing marine trash [42]. The authors used Mask R-CNN [121] and trained this structure to perform instance segmentation in the images, identifying objects (e.g., cup, bag, bottle) and materials (e.g., plastic, wood).

Still using simple models, DeepLabV3+ [122] and SegNet [123] were used to segment underwater images between background and foreground [44]. For training SegNet, the researchers initialized the weights randomly, and for training DeepLabV3+, they used Xcpetion backbone weights. The authors tried to augment the dataset using images from an indoor dataset with quality degradation to simulate the backscattering effect present in underwater images. However, using this strategy to augment the dataset harmed the model's performance. The models were evaluated in a test set of 300 real underwater images. DeepLabV3+ presented better much results when pre-training with PASCAL VOC [124], and Segnet presented slightly better results with no pre-training. This last observation highlights the importance of using transfer learning when training very deep neural networks with relatively small datasets. The authors also tested pre-processing the images using UGAN [62] and the UDCP method. However, pre-processing with UDCP [125] decreased the performance of the segmentation, and UGAN [62] only improved the performance of DeepLab.

The segment anything model (SAM) [126] has recently garnered attention for its easy adaptability to various segmentation tasks and high performance. AquaSAM is a model generated by automatically extracting labels from the SUIM dataset and fine-tuning SAM for this data [127]. AquaSAM outperforms SAM in almost all SUIM classes.

B. Structures inspired in well-known models

Other works tried modifying classic CNNs to improve performance or lower inference time.

A modified Mask R-CNN was used to segment pipelines and pipeline oil leakage in underwater images [128]. The authors used an improved filter pyramid that improves the information flow between high and low-level feature maps. They also modified the loss function, adding the boundary-weighted loss function. Both modifications increase the ability to segment the edge regions of the objects.

SUIM-Net is a model developed to be fast during the inference phase [43]. The authors used the strategies of encoder-decoder and skip connections between both. In the encoder, they added skip connections inside residual inspired blocks. The resulting model is faster than well-known models such as U-net [61] and DeepLabV3 [129]. Even though their results for mIoU and dice coefficient were worse than those achieved by classic models, they were still competitive.

For dealing with the problem of color degradation and low definition of objects in underwater images, two preprocessing modules were used before a SegNet structure [130]. These modules are rule-based and aim to stretch the color channels and extract edges with the Sobel filter. These modules are applied in parallel, and their results are fused and applied to SegNet. Finally, the output of SegNet is passed through a pyramid pooling module to achieve a better contextual understanding.

For extracting better features in the encoder, a main ResNet backbone was used with an auxiliary GhostConv CNN [131]. The authors also used a multi-scale feature fusion with channel attention in the decoder for not loosing information of deeper or shallower feature maps, and a mixed cross entropy and dice loss, for improving the boundaries segmentation.

C. Dealing with the lack of data

An adapted U-net [61] was used to deal with the problem of lack of data, using an adversarial transfer learning approach to train the model with ground crack and underwater crack images to learn to predict underwater dam cracks [132]. An attention module was used to get the model's attention to regions with cracks and decrease the influence of noise in the image. Taking a step further to solve the problem of labeled data, unsupervised training was used to teach w-net [133], a model composed of two U-net [61] structures in sequence, to segment images from fish4knowledge [134]. The first U-net is supposed to segment the images, and the second one should reconstruct the image. Soft normalized cut loss and reconstruction loss are used for the unsupervised training. Using a third approach, SegNet [123] was trained with photorealistic images and was then applied to segment real-world images between background and structures containing biofouling, achieving encouraging results [135].

A different strategy for saving annotation time is to start with sparse labels. For segmenting images with different coral species, an initial set of 100 annotation points per image was used to generate dense masks by clusterization [136]. The masks were used to train DeepLabv3+ [122]. Other similar works used DeepLabv3, DeepLabv3+ [137], and SegNet [138].

D. Image segmentation discussion

The research above shows that deep learning models can provide good results in segmenting underwater images. Furthermore, the models have many similarities with the classification and detection structures. These similarities encourage, for example, the use of spatial and channel attention modules used by the previous structures. Finally, different approaches to dealing with the lack of data seem like a promising research direction. For example, instead of first clustering the images using the sparse labels and then training a deep learning model, an option could be to adapt the loss of the segmentation model to train everything at once.

VIII. Discussion

This session discusses points identified during this survey that could be addressed by future research to allow faster development of computer vision with deep learning in the underwater environment.

Related work of this study revealed (1) the lack of details about the training method and the models and (2) the non-standardized strategies to present the results and comparisons.

Often multiple modifications are introduced simultaneously, complicating the identification of their individual impact. Additionally, varying datasets and base models in each paper hinder direct comparisons. The following suggestions would accelerate the development in this field:

1) Provide the details about the models allowing re-implementation;
2) Standardized datasets for each task would greatly benefit the community.
    - For classification: DeepFish;
    - For detection: UDD;
    - For semantic segmentation: SUIM;
    - For instance segmentation: TrashCan
3) The chosen data augmentation methods should be clearly specified, and the models should be evaluated with and without them.
4) Always start developing and testing the models with the raw images and then use the desired enhancement techniques, comparing both results.
5) Changes of the model should be evaluated individually. By extension multiple modifications should be introduced and evaluated one by one.
6) For metrics that can be calculated in more than one way, such as mAP, clearly explain what was used.
7) Use the same hyper-parameters to train all models, making the comparison fair;
8) Clearly identify the subsets of the data used for training, validation, and testing, allowing the repeatability of the results.
9) If applicable, Cross-validation should be reported in a reproducible fashion. When comparing and interpreting models it must be taken into account that different training runs yield different results.

The authors strongly believe that the measures above could benefit the development of computer vision with deep learning models in the underwater environment.

An important future direction of research is *overconfidence*. Deep learning models are referred to as overconfident if they assign high probability to to data which is out-of-distribution. For solving this overconfidence problem, the predictive uncertainty [139] of the models has been studied in critical fields, such as medical images [140]–[143]. However, for the underwater environment, the application of uncertainty in image analysis is new, appearing in the recent research about training segmentation models with active learning for segmenting pipeline images [144], overcoming the sim-to-real gap for models trained with synthetic underwater images [21], regularization to count fish in sonar image [145] and to perform classification of heterogeneous underwater soundscapes [146]. Using uncertainty to enhance the reliability of models in underwater environments is a research area that has not yet gained significant attention and should be further investigated.

IX. Conclusion

The studies referenced in this survey paper indicate that deep learning techniques have great potential to be used to classify, detect, and segment underwater images. However, it also points out the image quality challenges that differ from in-air images and jeopardize auxiliary strategies such as transfer learning and fine-tuning. Furthermore, underwater images have many overlapped objects and objects of different scales, often pointed out as challenging for deep learning models to deal with. As shown in Sect. VI, strategies such as using skip connections, applying dilation convolution, and using feature maps of different scales for the detection module present good results.

Regarding the lack of datasets, domain adaptation with GANs appears to be a promising solution when collecting images is not an option. Conversely, performing image enhancement before deep learning techniques should be further studied. It is not questionable that enhancing the images has visual improvements. However, its ability to improve the classification, detection, and segmentation performances seems tiny in many cases. The trade-off between the benefits and the effort of using enhancement methods should be further studied.

Comparing the different strategies used by researchers working with computer vision for the underwater environment is very difficult. Contrary to in-air studies that benchmark the developed models with vastly used datasets, there is no established standard dataset for the underwater scenario. This poses a challenge when comparing results from different studies in this domain and identify the more efficient strategies. A joint effort of the computer vision community working with the underwater domain should be taken to standardize the way of evaluating their achievements.

References

[1] C.-C. Wang and H. Samani, "Object detection using transfer learning for underwater robot," in *2020 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*. IEEE, 2020, pp. 1–4.

[2] O. Beijbom, P. J. Edmunds, D. I. Kline, B. G. Mitchell, and D. Kriegman, "Automated annotation of coral reef survey images," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1170–1177.

[3] M. Bax, D. Short, and D. Short, *Underwater inspection*. CRC Press, 1988.

[4] B. Jähne, H. Haussecker, and P. Geissler, *Handbook of computer vision and applications*. Citeseer, 1999, vol. 2.

[5] A. Rosenfeld, "Computer vision: basic principles," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 863–868, 1988.

[6] R. Cipolla, S. Battiato, G. M. Farinella *et al.*, *Machine learning for computer vision*. Springer, 2013, vol. 5.

[7] R. I. Hartley and J. L. Mundy, "Relationship between photogrammmetry and computer vision," *Integrating photogrammetric techniques with scene analysis and machine vision*, vol. 1944, pp. 92–105, 1993.

[8] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International journal of Remote sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[9] M. A. Abu, N. H. Indra, A. Rahman, N. A. Sapiee, and I. Ahmad, "A study on image classification based on deep learning and tensorflow," *Int. J. Eng. Res. Technol*, vol. 12, no. 4, pp. 563–569, 2019.

[10] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International journal of computer vision*, vol. 128, no. 2, pp. 261–318, 2020.

[11] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[12] K.-S. Fu and J. Mui, "A survey on image segmentation," *Pattern recognition*, vol. 13, no. 1, pp. 3–16, 1981.

[13] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[14] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–36, 2018.

[15] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019.

[16] F. Zhu, L. Shao, J. Xie, and Y. Fang, "From handcrafted to learned representations for human action recognition: A survey," *Image and Vision Computing*, vol. 55, pp. 42–52, 2016.

[17] L. Nanni, S. Ghidoni, and S. Brahnam, "Handcrafted vs. non-handcrafted features for computer vision classification," *Pattern Recognition*, vol. 71, pp. 158–172, 2017.

[18] J. Ma, X. Jiang, A. Fan, J. Jiang, and J. Yan, "Image matching from handcrafted to deep features: A survey," *International Journal of Computer Vision*, vol. 129, no. 1, pp. 23–79, 2021.

[19] Z. Alyafeai and L. Ghouti, "A fully-automated deep learning pipeline for cervical cancer classification," *Expert Systems with Applications*, vol. 141, p. 112951, 2020.

[20] C. P. Langlotz, B. Allen, B. J. Erickson, J. Kalpathy-Cramer, K. Bigelow, T. S. Cook, A. E. Flanders, M. P. Lungren, D. S. Mendelson, J. D. Rudie *et al.*, "A roadmap for foundational research on artificial intelligence in medical imaging: from the 2018 nih/rsna/acr/the academy workshop," *Radiology*, vol. 291, no. 3, p. 781, 2019.

[21] L. Ribeiro Marnet, Y. Brodskiy, S. Grasshof, and A. Wąsowski, "Bridging the sim-to-real gap for underwater image segmentation," in *OCEANS 2024-Singapore*. IEEE, 2024.

[22] O. Álvarez Tuñón, L. R. Marnet, L. Antal, M. Aubard, M. Costa, and Y. Brodskiy, "Subpipe: A submarine pipeline inspection dataset for segmentation and visual-inertial localization," in *OCEANS 2024-Singapore*. IEEE, 2024.

[23] R. Schettini and S. Corchs, "Underwater image processing: state of the art of restoration and image enhancement methods," *EURASIP journal on advances in signal processing*, vol. 2010, pp. 1–14, 2010.

[24] A. Duarte, F. Codevilla, J. D. O. Gaya, and S. S. C. Botelho, "A dataset to evaluate underwater image restoration methods," in *OCEANS 2016 - Shanghai*. IEEE, 2016, pp. 1–6.

[25] J. Y. Chiang and Y.-C. Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE transactions on image processing*, vol. 21, no. 4, pp. 1756–1769, 2011.

[26] M. Moniruzzaman, S. M. S. Islam, M. Bennamoun, and P. Lavery, "Deep learning on underwater marine object detection: A survey," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2017, pp. 150–160.

[27] B. Teng and H. Zhao, "Underwater target recognition methods based on the framework of deep learning: A survey," *International Journal of Advanced Robotic Systems*, vol. 17, no. 6, p. 1729881420976307, 2020.

[28] P. Sarkar, S. De, and S. Gurung, "A survey on underwater object detection," in *Intelligence Enabled Research*. Springer, 2022, pp. 91–104.

[29] S. Mittal, S. Srivastava, and J. P. Jayanth, "A survey of deep learning techniques for underwater image classification," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2022.

[30] M. J. Er, J. Chen, Y. Zhang, and W. Gao, "Research challenges, recent advances, and popular datasets in deep learning-based underwater marine object detection: A review," *Sensors*, vol. 23, no. 4, p. 1990, 2023.

[31] N. Wang, Y. Wang, and M. J. Er, "Review on deep learning techniques for marine object recognition: Architectures and algorithms," *Control Engineering Practice*, vol. 118, p. 104458, 2022.

[32] "Google search api." [Online]. Available: https://serpapi.com/

[33] M. J. Islam, Y. Xia, and J. Sattar, "Fast underwater image enhancement for improved visual perception," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 02 2020.

[34] M. C. R. LTER and P. Edmunds, "Mcr lter: Coral reef: Computer vision: Moorea labeled corals ver 3," Environmental Data Initiative, 2019, accessed 2024-07-19. [Online]. Available: https://doi.org/10.6073/pasta/88dde0e68ab5232a470389f4bedd1892

[35] M. Bewley, A. Friedman, R. Ferrari Legorreta, N. Hill, R. Hovey, N. Barrett, O. Pizarro, W. Figueira, L. Meyer, R. Babcock, L. Bellchambers, M. Byrne, and S. Williams, "Australian sea-floor survey data, with images and expert annotations," *Scientific Data*, vol. 2, 10 2015.

[36] G. Cutter, K. Stierhoff, and J. Zeng, "Automated detection of rockfish in unconstrained underwater videos using haar cascades and a new image dataset: Labeled fishes in the wild," *Proceedings - 2015 IEEE Winter Conference on Applications of Computer Vision Workshops, WACVW 2015*, pp. 57–62, 02 2015.

[37] M. Jian, Q. Qi, J. Dong, Y. Yin, W. Zhang, and K.-M. Lam, "The ouc-vision large-scale underwater image database," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 2017, pp. 1297–1302.

[38] M. Pedersen, J. Bruslund Haurum, R. Gade, and T. B. Moeslund, "Detection of marine animals in a new underwater dataset with varying visibility," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 18–26.

[39] C. Liu, Z. Wang, S. Wang, T. Tang, Y. Tao, C. Yang, H. Li, X. Liu, and X. Fan, "A new dataset, poisson gan and aquanet for underwater object grabbing," 2021.

[40] C. Liu, H. Li, S. Wang, M. Zhu, D. Wang, X. Fan, and Z. Wang, "A Dataset And Benchmark Of Underwater Object Detection For Robot Picking," *arXiv e-prints*, p. arXiv:2106.05681, Jun. 2021.

[41] K. Katija, B. Schlining, L. Lundsten, K. Barnard, G. Sainz, O. Boulais, B. Woodward, and K. C. Bell, "Fathomnet: An open, underwater image repository for automated detection and classification of midwater and benthic objects," *Marine Technology Society Journal*, vol. 55, no. 3, pp. 136–137, 2021.

[42] J. Hong, M. Fulton, and J. Sattar, "Trashcan: A semantically-segmented dataset towards visual detection of marine debris," *arXiv preprint arXiv:2007.08097*, 2020.

[43] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1769–1776.

[44] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas, and S. S. C Botelho, "Underwater image segmentation in the wild using deep learning," *Journal of the Brazilian Computer Society*, vol. 27, no. 1, pp. 1–14, 2021.

[45] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European conference on computer vision*. Springer, 2012, pp. 746–760.

[46] M. Waszak, A. Cardaillac, B. Elvesæter, F. Rødølen, and M. Ludvigsen, "Semantic segmentation in underwater ship inspections: Benchmark and data set," *IEEE Journal of Oceanic Engineering*, vol. 48, no. 2, pp. 462–473, 2022.

[47] A. Saleh, I. H. Laradji, D. A. Konovalov, M. Bradley, D. Vazquez, and M. Sheaves, "A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis," *Scientific Reports*, vol. 10, no. 1, p. 14671, 2020.

[48] C. Li, S. Anwar, and F. Porikli, "Underwater scene prior inspired deep underwater image and video enhancement," *Pattern Recognition*, vol. 98, p. 107038, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320319303401

[49] J. Perez, A. C. Attanasio, N. Nechyporenko, and P. J. Sanz, "A deep learning approach for underwater image enhancement," in *Biomedical Applications Based on Natural and Artificial Computing*, J. M. Ferrández Vicente, J. R. Álvarez-Sánchez, F. de la Paz López, J. Toledo Moreo, and H. Adeli, Eds. Cham: Springer International Publishing, 2017, pp. 183–192.

[50] G. Ramkumar, A. G, S. K. M, M. Ayyadurai, and S. C, "An effectual underwater image enhancement using deep learning algorithm," in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2021, pp. 1507–1511.

[51] X. Chen, P. Zhang, L. Quan, C. Yi, and C. Lu, "Underwater image enhancement based on deep learning and image formation model," 2021.

[52] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–, 11 2004.

[53] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, 1981.

[54] E. H. Land, "The retinex theory of color vision," *Scientific American*, vol. 237, no. 6, pp. 108–129, 1977. [Online]. Available: http://www.jstor.org/stable/24953876

[55] C. Wei, W. Wang, W. Yang, and J. Liu, "Deep retinex decomposition for low-light enhancement," *CoRR*, vol. abs/1808.04560, 2018. [Online]. Available: http://arxiv.org/abs/1808.04560

[56] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater image processing and object detection based on deep cnn method," *J. Sensors*, vol. 2020, 2020. [Online]. Available: https://doi.org/10.1155/2020/6707328

[57] S. Cao, D. Zhao, Y. Sun, and C. Ruan, "Learning-based low-illumination image enhancer for underwater live crab detection," *ICES Journal of Marine Science*, vol. 78, no. 3, pp. 979–993, 01 2021. [Online]. Available: https://doi.org/10.1093/icesjms/fsaa250

[58] C.-H. Yeh, C.-H. Huang, and C.-H. Lin, "Deep learning underwater image color correction and contrast enhancement based on hue preservation," in *2019 IEEE Underwater Technology (UT)*, 2019, pp. 1–6.

[59] T. Katayama, T. Song, T. Shimamoto, and X. Jiang, "Gan-based color correction for underwater object detection," in *OCEANS 2019 MTS/IEEE SEATTLE*, 2019, pp. 1–4.

[60] L. Chen, Z. Jiang, L. Tong, Z. Liu, A. Zhao, Q. Zhang, J. Dong, and H. Zhou, "Perceptual underwater image enhancement with deep learning and physical priors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, pp. 3078–3092, 2021.

[61] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[62] C. Fabbri, M. J. Islam, and J. Sattar, "Enhancing underwater imagery using generative adversarial networks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7159–7165.

[63] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016.

[64] C. Edge, M. J. Islam, C. Morse, and J. Sattar, "A generative approach for detection-driven underwater image enhancement," *CoRR*, vol. abs/2012.05990, 2020. [Online]. Available: https://arxiv.org/abs/2012.05990

[65] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007%2F978-3-319-46448-0_2

[66] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[67] X. Fu, P. Zhuang, Y. Huang, Y. Liao, X.-P. Zhang, and X. Ding, "A retinex-based enhancing approach for single underwater image," in *2014 IEEE international conference on image processing (ICIP)*. Ieee, 2014, pp. 4572–4576.

[68] Z. Liu, B. Wang, Y. Li, J. He, and Y. Li, "Unitmodule: A lightweight joint image enhancement module for underwater object detection," *Pattern Recognition*, vol. 151, p. 110435, 2024.

[69] J. Zhang, L. Zhu, L. Xu, and Q. Xie, "Research on the correlation between image enhancement and underwater object detection," in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 5928–5933.

[70] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[71] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, 01 2012.

[72] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[73] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[74] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[75] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[76] Z. Cao, J. C. Principe, B. Ouyang, F. Dalgleish, and A. Vuorenkoski, "Marine animal classification using combined cnn and hand-designed image features," in *OCEANS 2015-MTS/IEEE Washington*. IEEE, 2015, pp. 1–6.

[77] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[78] A. Mahmood, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "Resfeats: Residual network based features for underwater image classification," *Image and Vision Computing*, vol. 93, p. 103811, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885619301362

[79] V. Lopez-Vazquez, J. M. Lopez-Guede, S. Marini, E. Fanelli, E. Johnsen, and J. Aguzzi, "Video image enhancement and machine learning pipeline for underwater animal detection and classification at cabled observatories," *Sensors*, vol. 20, no. 3, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/3/726

[80] L. Jin and H. Liang, "Deep learning for underwater image recognition in small sample size situations," in *OCEANS 2017 - Aberdeen*, 2017, pp. 1–4.

[81] P. Szymak, P. Piskur, and K. Naus, "remote sensing the effectiveness of using a pretrained deep learning neural networks for object classification in underwater video," *Remote Sensing*, vol. 12, 09 2020.

[82] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[83] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[84] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[85] C. Chen, W. Xie, Y. Huang, X. Yu, and X. Ding, "Weakly-supervised man-made object recognition in underwater optimal image through deep domain adaptation," in *International Conference on Neural Information Processing*. Springer, 2018, pp. 311–322.

[86] Y. Xu, Y. Zhang, H. Wang, and X. Liu, "Underwater image classification using deep convolutional neural networks and data augmentation," in *2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2017, pp. 1–5.

[87] C.-C. Wang, H. Samani, and C.-Y. Yang, "Object detection with deep learning for underwater environment," in *2019 4th International Conference on Information Technology Research (ICITR)*, 2019, pp. 1–6.

[88] P. Athira., T. Mithun Haridas, and M. Supriya, "Underwater object detection model based on yolov3 architecture using deep neural networks," in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, 2021, pp. 40–45.

[89] M. Asyraf, I. Isa, M. Marzuki, S. Sulaiman, and C. Hung, "Cnn-based yolov3 comparison for underwater object detection," *Journal of Electrical & Electronic Systems Research*, vol. 18, pp. 30–37, 04 2021.

[90] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018. [Online]. Available: https://arxiv.org/abs/1804.02767

[91] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[92] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[93] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, ser. Adaptive computation and machine learning. The MIT Press, 2016.

[94] F. Chollet, "The keras blog," Jan 2016. [Online]. Available: https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html

[95] H. Yuan, S. Zhang, G. Chen, and Y. Yang, "Underwater Image Fish Recognition Technology Based on Transfer Learning and Image Enhancement," *Journal of Coastal Research*, vol. 105, no. sp1, pp. 124 – 128, 2020. [Online]. Available: https://doi.org/10.2112/JCR-SI105-026.1

[96] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[97] L. Chen, F. Zhou, S. Wang, J. Dong, N. Li, H. Ma, X. Wang, and H. Zhou, "SWIPENET: object detection in noisy underwater images," *CoRR*, vol. abs/2010.10006, 2020. [Online]. Available: https://arxiv.org/abs/2010.10006

[98] L. Chen, Z. Liu, L. Tong, Z. Jiang, S. Wang, J. Dong, and H. Zhou, "Underwater object detection using invert multi-class adaboost with deep learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[99] Z. Jiang and R. Wang, "Underwater object detection based on improved single shot multibox detector," in *2020 3rd International Conference on Algorithms, Computing and Artificial Intelligence*, ser. ACAI 2020. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3446132.3446170

[100] W. Xu and S. Matzner, "Underwater fish detection using deep learning for water power applications," *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 313–318, 2018.

[101] A. F. Ayob, K. Khairuddin, Y. M. Mustafah, A. R. Salisa, and K. Kadir, "Analysis of pruned neural networks (mobilenetv2-yolo v2) for underwater object detection," in *Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019*, Z. Md Zain, H. Ahmad, D. Pebrianti, M. Mustafa, N. R. H. Abdullah, R. Samad, and M. Mat Noh, Eds. Singapore: Springer Singapore, 2021, pp. 87–98.

[102] B. Fan, W. Chen, Y. Cong, and J. Tian, "Dual refinement underwater object detection network," in *ECCV*, 2020.

[103] S. Liu, D. Huang *et al.*, "Receptive field block net for accurate and fast object detection," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 385–400.

[104] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[105] W.-H. Lin, J.-X. Zhong, S. Liu, T. Li, and G. Li, "Roimix: proposal-fusion among multiple images for underwater object detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 2588–2592.

[106] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[107] T.-S. Pan, H.-C. Huang, J.-C. Lee, and C.-H. Chen, "Multi-scale resnet for real-time underwater object detection," *Signal, Image and Video Processing*, vol. 15, pp. 1–9, 07 2021.

[108] L. Zhang, X. Yang, Z. Liu, L. Qi, H. Zhou, and C. Chiu, "Single shot feature aggregation network for underwater object detection," *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 1906–1911, 2018.

[109] H. Liu, P. Song, and R. Ding, "Towards domain generalization in underwater object detection," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 1971–1975.

[110] L. Chen, J. Dong, and H. Zhou, "Class balanced underwater object detection dataset generated by class-wise style augmentation," 2021.

[111] J. Zhang, L. Zhu, L. Xu, and Q. Xie, "Mffssd: An enhanced ssd for underwater object detection," in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 5938–5943.

[112] W. Chen and B. Fan, "Underwater object detection with mixed attention mechanism and multi-enhancement strategy," in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 2821–2826.

[113] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[114] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 713–13 722.

[115] W. Ji, J. Peng, B. Xu, and T. Zhang, "Real-time detection of underwater river crab based on multi-scale pyramid fusion image enhancement and mobilecenternet model," *Computers and Electronics in Agriculture*, vol. 204, p. 107522, 2023.

[116] M. Gao, S. Li, K. Wang, Y. Bai, Y. Ding, B. Zhang, N. Guan, and P. Wang, "Real-time jellyfish classification and detection algorithm based on improved yolov4-tiny and improved underwater image enhancement algorithm," *Scientific Reports*, vol. 13, no. 1, p. 12989, 2023.

[117] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[118] D. Zhang, L. Li, Z. Zhu, S. Jin, W. Gao, and C. Li, "Object detection algorithm based on deformable convolutional networks for underwater images," in *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)*, 2019, pp. 274–279.

[119] X. Lv, A. Wang, Q. Liu, J. Sun, and S. Zhang, "Proposal-refined weakly supervised object detection in underwater images," in *Image and Graphics: 10th International Conference, ICIG 2019, Beijing, China, August 23–25, 2019, Proceedings, Part I 10*. Springer, 2019, pp. 418–428.

[120] L. Thampi, R. Thomas, S. Kamal, A. A. Balakrishnan, T. M. Haridas, and M. Supriya, "Analysis of u-net based image segmentation model on underwater images of different species of fishes," in *2021 International Symposium on Ocean Technology (SYMPOL)*. IEEE, 2021, pp. 1–5.

[121] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[122] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[123] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1511.00561

[124] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[125] P. L. Drews, E. R. Nascimento, S. S. Botelho, and M. F. M. Campos, "Underwater depth estimation and image restoration based on single images," *IEEE computer graphics and applications*, vol. 36, no. 2, pp. 24–35, 2016.

[126] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[127] M. Xu, J. Su, and Y. Liu, "Aquasam: Underwater image foreground segmentation," *arXiv preprint arXiv:2308.04218*, 2023.

[128] X. Zhao, L. Jing, and Z. Du, "Research on image segmentation method of underwater pipeline oil leakage point," in *2020 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, 2020, pp. 165–170.

[129] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[130] J. Wang, F. Zha, and X. Bi, "Research on underwater image semantic segmentation method based on segnet," in *Proceedings of the International Conference of Fluid Power and Mechatronic Control Engineering (ICFPMCE 2022)*, vol. 10. Springer Nature, 2023, p. 131.

[131] Z. He, L. Cao, J. Luo, X. Xu, J. Tang, J. Xu, G. Xu, and Z. Chen, "Uiss-net: Underwater image semantic segmentation network for improving boundary segmentation accuracy of underwater images," *Aquaculture International*, pp. 1–14, 2024.

[132] X. Fan, P. Cao, P. Shi, X. Chen, X. Zhou, and Q. Gong, "An underwater dam crack image segmentation method based on multi-level adversarial transfer learning," *Neurocomputing*, vol. 505, pp. 19–29, 2022.

[133] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," 2017. [Online]. Available: https://arxiv.org/abs/1711.08506

[134] E. S. Saleh, T. M. Haridas, and M. Supriya, "Unsupervised image segmentation model based on w net architecture and conditional random field for underwater images," in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1. IEEE, 2021, pp. 34–39.

[135] M. O'Byrne, V. Pakrashi, F. Schoefs, and B. Ghosh, "Semantic segmentation of underwater imagery using deep networks trained on synthetic imagery," *Journal of Marine Science and Engineering*, vol. 6, no. 3, p. 93, 2018.

[136] S. Raine, R. Marchant, B. Kusy, F. Maire, and T. Fischer, "Point label aware superpixels for multi-species segmentation of underwater imagery," *arXiv e-prints*, pp. arXiv–2202, 2022.

[137] I. Alonso, M. Yuval, G. Eyal, T. Treibitz, and A. C. Murillo, "Coralseg: Learning coral segmentation from sparse annotations," *Journal of Field Robotics*, vol. 36, no. 8, pp. 1456–1477, 2019.

[138] I. Alonso, A. Cambra, A. Munoz, T. Treibitz, and A. C. Murillo, "Coral-segmentation: Training dense labeling models with sparse ground truth," in *Proceedings of the IEEE international conference on computer vision workshops*, 2017, pp. 2874–2882.

[139] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[140] A. Filos, S. Farquhar, A. N. Gomez, T. G. J. Rudner, Z. Kenton, L. Smith, M. Alizadeh, A. de Kroon, and Y. Gal, "A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks," 2019. [Online]. Available: https://arxiv.org/abs/1912.10481

[141] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl, "Leveraging uncertainty information from deep neural networks for disease detection," *Scientific reports*, vol. 7, no. 1, pp. 1–14, 2017.

[142] A. Jungo, R. Meier, E. Ermis, E. Herrmann, and M. Reyes, "Uncertainty-driven sanity check: Application to postoperative brain tumor cavity segmentation," 2018. [Online]. Available: https://arxiv.org/abs/1806.03106

[143] A. Jungo, R. McKinley, R. Meier, U. Knecht, L. Vera, J. Pérez-Beteta, D. Molina-García, V. M. Pérez-García, R. Wiest, and M. Reyes, "Towards uncertainty-assisted brain tumor segmentation and survival prediction," in *International MICCAI Brainlesion Workshop*. Springer, 2017, pp. 474–485.

[144] L. Ribeiro Marnet, Y. Brodskiy, S. Grasshof, and A. Wąsowski, "Uncertainty driven active learning for image segmentation in underwater inspection," in *International Conference on Robotics, Computer Vision and Intelligent Systems*. Springer, 2024, pp. 66–81.

[145] P. Tarling, M. Cantor, A. Clapés, and S. Escalera, "Deep learning with self-supervision and uncertainty regularization to count fish in underwater images," *PloS one*, vol. 17, no. 5, p. e0267759, 2022.

[146] B. Beckler, A. Pfau, M. Orescanin, S. Atchley, N. Villemez, J. E. Joseph, C. W. Miller, and T. Margolina, "Multilabel classification of heterogeneous underwater soundscapes with bayesian deep learning," *IEEE Journal of Oceanic Engineering*, 2022.

Appendix B

Paper - MIMIR-UW: A Multipurpose Synthetic Dataset for Underwater Navigation and Inspection

MIMIR-UW: A Multipurpose Synthetic Dataset for Underwater Navigation and Inspection

Olaya Álvarez-Tuñón, Hemanth Kanner, Luiza Ribeiro Marnet, Huy Xuan Pham, Jonas le Fevre Sejersen, Yury Brodskiy, and Erdal Kayacan

***Abstract*— This paper presents MIMIR-UW, a multipurpose underwater synthetic dataset for SLAM, depth estimation, and object segmentation to bridge the gap between theory and application in underwater environments. MIMIR-UW integrates three camera sensors, inertial measurements, and ground truth for robot pose, image depth, and object segmentation. The underwater robot is deployed within a pipe exploration scenario, carrying artificial lights that create uneven lighting, in addition to natural artefacts such as reflections from natural light and backscattering effects. Four environments totalling eleven tracks are provided, with various difficulties regarding light conditions or dynamic elements. Two metrics for dataset evaluation are proposed, allowing MIMIR-UW to be compared with other datasets. State-of-art methods on SLAM, segmentation and depth estimation are deployed and benchmarked on MIMIR-UW. Moreover, the dataset's potential for sim-to-real transfer is demonstrated by leveraging the segmentation and depth estimation models trained on MIMIR-UW in a real pipeline inspection scenario. To the best of the authors' knowledge, this is the first underwater dataset targeted for such a variety of methods. The dataset is publicly available online. `https://github.com/remaro-network/MIMIR-UW/`**

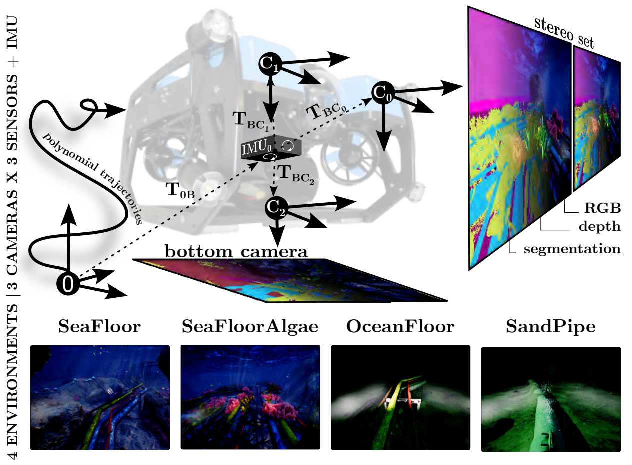


Fig. 1. MIMIR-UW dataset gathering pipeline: a camera rig consisting of a stereo set (C_0,C_1) and a downward-looking camera (C_2) recording RGB, depth, and segmentation in four underwater environments. The robot follows polynomial trajectories with varying speeds and accelerations.

I. Introduction

While simulators and datasets are critical in developing computer vision algorithms, multiple factors have limited their availability in underwater scenarios. For instance, the strong attenuation under water hinders the use of state-of-art ground truth pose and motion capture devices. Moreover, generating ground truth for segmentation requires specialization on the subjects, such as coral, fish species, and pipeline damage. The accessibility of deployment areas is another critical down-weighting factor. Offshore structures are suitable for underwater robot deployment, as they require regular inspections [1]. However, in the oil and gas industries, the principal owners are generally unwilling to open their data for public use for privacy and security reasons.

Unlike geometry-based localization approaches, learning-based localization algorithms are affected by the diversity of imaging conditions and the camera's motion patterns. Similarly, other learning methods, such as segmentation and depth estimation, require diverse imaging conditions to achieve generalization. Prevailing datasets for simultaneous localization and mapping (SLAM), segmentation, and depth estimation have been primarily gathered with ground or aerial devices, which are constrained by their model's dynamics and environmental conditions. Bringing deep learning techniques for computer vision to the underwater environment requires the availability of datasets in such conditions.

Simulators are a viable alternative for dataset generation under the mentioned difficulties for data collection inherent to underwater environments. Realistic simulation renderings have made sim-to-real transfer a fundamental discipline in robotics [13], [14]. However, realistic simulations of underwater imaging conditions [15], [16] have yet to be incorporated into the prevailing open-source robotic simulators [17]–[19]. Nevertheless, the open-source 3D generation tool Unreal Engine provides a framework with realistic image renderings, which, integrated with external plugins [20], [21], provides frameworks for robotics simulation. The AirSim plugin [21] provides an API for collecting pose, segmentation, and depth estimation ground truth, along with Robot Operating System (ROS) integration. This paper proposes an underwater synthetic dataset collected with Unreal Engine and AirSim, in the context of pipeline inspection (See Fig. 1). The contributions of this paper are:

1) A synthetic dataset for localization, segmentation, and depth estimation. It has been recorded under four underwater scenarios, with varying trajectories, lighting

O. Álvarez, H. X. Pham, J. Le Fevre are with Artificial Intelligence in Robotics Laboratory (AiRLab), the Department of Electrical and Computer Engineering, Aarhus University, 8000 Aarhus C, Denmark {`olaya, huy.pham, jonas.le.fevre`} `at ece.au.dk`. H. Kanner, L. Ribeiro and Y. Brodskiy are with EIVA a/s, 8660 Skanderborg, Denmark {`hek,lrm,ybr`} `at eiva.com` E. Kayacan is with Automatic Control Group (RAT), Paderborn University, 33098 Paderborn, Germany {`erdal.kayacan at uni-paderborn.de`}.

TABLE I

Comparison of the state-of-the-art datasets for SLAM, segmentation and depth estimation purposes.

Dataset	Type	Environment	Pose	Segmentation	Camera	Depth	Perceptual aliasing	Dynamic scenes
KITTI [2]	real	city	✓	✓	stereo	✓	✗	✓
EuRoC [3]	real	indoor	✓	✗	stereo	✗	✗	✗
TUM-RGB-D [4]	real	indoor	✓	✗	stereo	✓	✓	✓
ETH-MS [5]	real	in/outdoor	✓	✗	rig	✗	✗	✗
Aqualoc [6]	real	underwater	✓	✗	monocular	✗	✓	✓
AURORA [7]	real	underwater	✓	✗	monocular	✗	✓	✓
Caves [8]	real	underwater	✓	✗	monocular	✓	✓	✓
TartanAIR [9]	synthetic	miscellaneous	✓	✓	monocular	✓	✗	✓
TrashCan [10]	real	underwater	✗	✓	monocular	✗	N/A	N/A
SUIM [11]	real	underwater	✗	✓	monocular	✗	N/A	N/A
NAUTEC UWI [12]	real	underwater	✗	✓	monocular	✗	N/A	N/A
MIMIR-UW (ours)	Synthetic	underwater	✓	✓	rig	✓	✓	✓

conditions, and presence of dynamic elements.

2) Introduction of the metrics that quantitatively describe characteristics of the dataset.
3) Experimental evaluation with state-of-art algorithms for SLAM, segmentation, and depth estimation. The dataset's capabilities for sim-to-real transfer of the learning-based segmentation and depth estimation methods are tested under a real-life pipe inspection scenario.

To the best of our knowledge, this is the first underwater dataset to introduce such a variety of labelled data and a pipeline inspection scenario. This way, we aim to ease algorithm testing and to push the development of learning-based computer vision algorithms through sim-to-real transfer for underwater environments.

The paper's outline is as follows: Section II introduces the related works on dataset gathering. Then, Section III presents the MIMIR-UW and the data collection process. Section IV proposes the metrics for dataset comparison, and Section V demonstrates the dataset's pertinence under the proposed baseline algorithms. Finally, some conclusions are drawn from this study in Section VI.

II. Related Works

Deep learning methods' high data requirements and the challenging conditions that the underwater environment poses for robot deployment have recently motivated the development of underwater robotics simulators and datasets. However, the simulation of imaging conditions is limited to exponential attenuation or sunlight reflections [15], [17], [18], making them insufficient for testing computer vision systems. Openly available datasets for underwater computer vision cover a variety of scopes, from object detection [22], [23], to image segmentation [10]–[12] and localization [6]–[8]. Ground truth retrieval for localization algorithms is particularly challenging underwater; hence the availability of datasets for visual localization is limited. Namely, Aqualoc dataset provides an estimated ground truth from the Colmap library [6]. Well-known datasets like EuRoC [3], KITTI [2], TUM-RGB-D [4], and ETH-MS [5] have been widely used for algorithm benchmarking. TartanAir [9] introduces a synthetic dataset with diverse environments presenting challenging imaging conditions. The difficulty levels of these datasets have been qualitatively described according to the presence of imaging defects caused by motion blur, illumination conditions, or dynamic elements. Table I compares MIMIR-UW and the mentioned datasets in accordance with these characteristics. MIMIR-UW aims to simulate imaging settings specific to the underwater environment, including perceptual aliasing, dynamic features, and other natural distortions. Additionally, compared to previous underwater datasets, MIMIR-UW comprises a broader range of ground truth data. The imaging fidelity with respect to other underwater datasets can be seen in Fig. 2.

III. MIMIR-UW - The multipurpose dataset

A. Data collection

The proposed dataset is collected from four underwater environments created in Unreal Engine 4, referred to as ”SeaFloor”, ”SeaFloor Algae”, ”OceanFloor”, and ”SandPipe”, in the context of pipeline inspection. Figure 2 depicts a sample of the recorded images. As shown in Table II, various tracks are recorded under each environment, with different lengths and durations. In all environments, the robot carries two artificial lights creating uneven light reflections throughout the scene. SeaFloor presents a shallow underwater environment, with good visibility but under the presence of dynamic light reflections from natural light. SeaFloor Algae extends the challenges in SeaFloor incorporating a higher concentration of dynamic objects in the scene occluding the pipes. OceanFloor is a deep underwater

TABLE II

Practical aspects about MIMIR-UW.

Environment	Sequence	Duration [s]	Path length [m]	#frames	#poses
SeaFloor	track0	120.840	238.623	2847	19927
	track1	89.693	191.460	2030	14204
	track2	109.157	244.969	2537	17754
SeaFloor Algae	track0	125.823	249.013	2934	20538
	track1	89.753	186.586	2076	14526
	track2	107.696	245.851	2489	17418
Ocean floor	track0_light	107.843	226.698	2421	16944
	track0_dark	107.603	227.482	2404	16825
	track1_light	279.132	714.786	6263	43837
Sandpipe	track0_dark	120.891	298.885	2741	19185
	track0_light	117.810	294.957	2605	18230

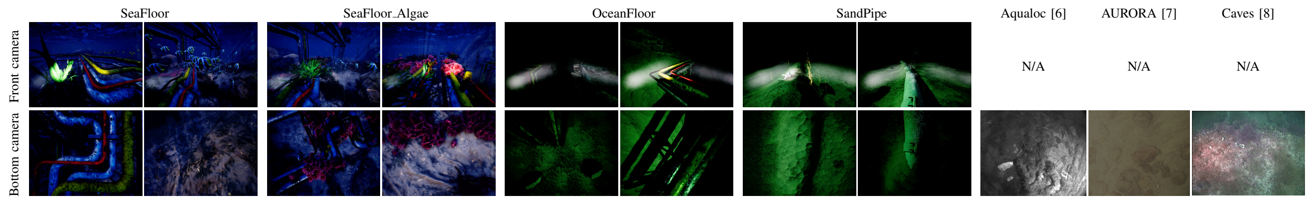


Fig. 2. Sample images for each of MIMIR-UW's environments. For recreating the underwater visual effects, the environments include; water distortion, floating particles, caustics, bubbles, fishes, rock formations, and reefs. In contrast to other underwater datasets like Aqualoc, Caves, and AURORA, MIMIR-UW provides a front camera view and pipe inspection elements. The absence of front camera images is indicated as not available (N/A).

environment where most visibility comes from the robot's artificial light. Similarly, SandPipe also integrates a deep underwater environment, but in this case, the pipe is covered by sand, with some exposed areas through the trajectory.

1) Sensor data and ground truth collection: The dataset is recorded with three cameras: two forward-looking cameras in a stereo configuration and a downward-looking camera. Each camera comprises three data sources: RGB, segmentation, and depth, gathered with AirSim's image API. The segmentation labels are automatically generated by AirSim's image API. Each RGB image from the camera has assigned a segmentation image, where each pixel has a color assigned to the object's class it belongs to. There is a total of fourteen classes corresponding to dynamic objects such as fishes, bubbles and algae, and static objects comprising pipes, rocks, the seafloor and the sky. The depth, as provided by AirSim's image API, is recorded as a floating point image containing the depth values in meters, with all the points in the plane parallel to the camera plane retrieving the same depth value. This depth recording is converted and stored as an inverse depth image for storage efficiency. The depth values recorded as zeros are stored as zeros in the inverse depth image to avoid zero division. AirSim's API also retrieves inertial measurements and pose ground truth, which are also provided in the dataset. For each track, we provide Airsim's settings file with which it is recorded and each camera's intrinsic and extrinsic parameters.

2) Trajectory generation: The dataset is recorded by a simulated underwater robot following trajectories generated from a set of waypoints. The waypoints are selected to generate trajectories that ensure motion diversity within the given setup and revisiting the same areas under different points of view for loop closure. The minimum-snap trajectory generation method generates smooth trajectories connecting waypoints [24]. They are generated as a 10-degree polynomial function describing the four-dimensional state, corresponding to the three linear axes and the robot heading with respect to time. For simplicity, tracking those trajectories is reduced to a general point-mass control problem.

IV. Dataset metrics

This section proposes a set of measurements to quantitatively evaluate the dataset's characteristics and compare them with existing datasets. The proposed metrics allow to gauge complexity of data for a certain class of algorithms, thus making algorithm benchmarking more uniform.

A. Image entropy

We rely on entropy to quantify the amount of information contained in each image. The image's spatial information is incorporated into Shannon's second-order entropy using the isotropic formulation derived in [25].

Consider a discrete band-limited signal f containing $2N \times 2M$ samples in the x and y axes, respectively. The signal's derivatives f_x and f_y are obtained by applying a convolution Sobel operator to the image. Then, the joint probability density function $p_{i,j}$ of the signal's derivative is computed using the Kronecker delta function $\delta_{i,j}$ as:

$$p_{i,j} = \frac{1}{4MN} \sum_{n=-N}^{N-1} \sum_{m=-M}^{M-1} \delta_{i,f_x(m,n)} \delta_{j,f_y(m,n)} \tag{1}$$

Hence, Shannon's joint entropy formulation yields the delentropy as:

$$H(f_x, f_y) = -\sum_{j=1}^{J} \sum_{i=1}^{I} p_{i,j} \log_2(p_{i,j}). \tag{2}$$

The delentropy value is maximized for a uniform distribution of $p_{i,j}$. This indicates the presence of nondistinctive features in the image, which can lead to misclassified pixels in segmentation, or mismatched features in feature-based localization. On the contrary, the entropy is minimized for zero-gradient images containing no information for the segmentation to classify or for the localization to track. Therefore, the ideal conditions for the proposed computer vision pipelines lie in intermediate values for delentropy.

B. Motion diversity

To assess the diversity of motion patterns in the dataset, the metric proposed in [9] is adopted, based on principal component analysis of the motion pattern. Given a sequence of n concatenated relative translations $\mathbf{T} \in \mathbb{R}^{3\times n}$ and rotations $\mathbf{R} \in \mathbb{R}^{4\times 4\times n}$ in $so(3)$, their principal motion components are computed through singular value decomposition. The obtained eigenvalues (t_1, t_2, t_3) and (r_1, r_2, r_3) and their associated eigenvectors represent the first, second and third principle motion axes of the sequence and their magnitudes.

The motion diversity metric is then obtained as:

$$\sigma = \frac{1}{2}\left(\frac{\sqrt{t_2 t_3}}{t_1} + \frac{\sqrt{r_2 r_3}}{r_1}\right), \tag{3}$$

where evenly distributed motions retrieve equal values for the eigenvalues, making the motion diversity metric converge to

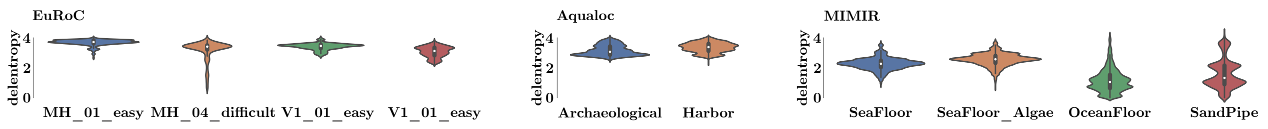


Fig. 3. Distribution of the delentropy values for the sequences in EuRoC, Aqualoc and MIMIR-UW. Compared to other datasets, MIMIR-UW provides a higher concentration of lower entropy values caused by blurred, dark, and low-contrast images.

one. Conversely, an absence of motion in one or both of the secondary axes yields a value of zero for its eigenvalues, and consequently, for the motion diversity metric.

C. Results

The dataset comparison candidates are EuRoC and Aqualoc; both of which present recordings from mobile platforms with degrees of freedom comparable to MIMIR-UW. Aqualoc comprises underwater sequences recorded in a real environment. Although EuRoC was recorded above water it is a well-established dataset, serving as a comparison of differences in the imaging conditions between underwater and above-water environments, and their impact on the proposed metrics. The delentropy results are shown in Fig. 3. EuRoC's difficult sequences, in contrast to the easy ones, present low values for delentropy generated by dark or blurred images. The delentropy values in EuRoC and Aqualoc are clustered around higher values than those retrieved by MIMIR-UW, considering that they present sequences with higher texture and contrast. MIMIR-UW's OceanFloor and SandPipe contain very dark images, where only the nearby elements in the scene are visible, and completely dark images without any nearby objects, resulting in a high amount of low delentropy values. SeaFloor Algae provides the highest entropy values for MIMIR-UW, originating from the algae's rich textures. Considering that the state-of-the-art algorithms proposed in Section V are intended for datasets richer in visual information, the delentropy metric provides a first impression of how the environments proposed by MIMIR-UW will challenge them.

Aqualoc provides the lowest motion diversity value since the mobile robot only moves in the xy plane (see Table III). MIMIR-UW's trajectories are designed to maximize diversity in the linear axes, outperforming EuRoC in the translation metric, and having both EuRoC and MIMIR-UW with similar motion diversity metrics as a total. However, EuRoC provides a higher diversity in rotation.

V. Experimental evaluation

This Section performs an experimental evaluation of the challenges and opportunities that MIMIR-UW presents for for SLAM, depth estimation, and object segmentation algorithms. Moreover, a sample dataset of a real-world pipeline inspection scenario (shown in Fig. 4) is used to demonstrate MIMIR-UW's capabilities for sim-to-real transfer. This sample dataset contains manually-generated labels for segmentation, and a sparse pseudo ground truth for depth generated by the photogrammetry framework AliceVision [26]. The real pipeline scenario is not publicly available, but sample images of ground truth and results are depicted in Figs. 5 and 6.

A. SLAM

To demonstrate the provided dataset's challenges for SLAM, we run ORB-SLAM3 [27] in both a monocular and stereo fashion. ORB-SLAM is one of the primary state-of-the-art indirect visual SLAM algorithms. One of the main causes of failure for previous versions of ORB-SLAM was track loss and subsequent relocalization failure. However, since ORB-SLAM3, the back-end integrates the so-called Atlas map: every time the tracking gets lost, a new "active" map is initialized, and the previous map is stored in memory, turning into a "passive" map. During runtime, the back-end looks for loop closures on both the active and the set of passive maps available. Considering that the tracks in MIMIR-UW are designed for loop closure detection, the Atlas map presents an interesting feature to analyze. On the other hand, the presence of scattering and low-texture areas in the dataset challenges indirect methods like ORB-SLAM, which relies on feature tracking. A family of methods that is of interest for such conditions is direct-based SLAM. Direct-based SLAM optimizes the aggregated photometric error around a parameter that can either be the camera transform or the inverse depth map. Thus, the direct-based SLAM algorithm DSO [28] is deployed, which achieves real-time performance by choosing the high-gradient pixels for the photometric error calculation.

The drift between estimated and ground-truth trajectories over the keyframe camera poses is evaluated with the absolute position error (APE) and the relative position error (RPE) [4]. The APE and the RPE quantify the global and local consistency of the trajectory, respectively. The

TABLE III
Motion diversity metric.

	Euroc			Aqualoc			MIMIR		
	translation	rotation	total	translation	rotation	total	translation	rotation	total
σ	0.334	**0.359**	**0.347**	0.226	0.067	0.147	**0.684**	0.008	0.345

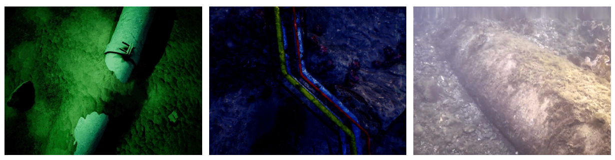


Fig. 4. From left to right, pipe images from SandPipe, SeaFloor, and the real scenario used for sim-to-real transfer. The shape, colour, and environmental conditions for SandPipe are more similar to this sample of real-world data.

TABLE IV
RESULTS OF DEPLOYING THE PROPOSED SLAM ALGORITHMS IN MIMIR-UW. THE BEST RESULTS FOR RELATIVE ERRORS ARE MARKED IN BOLD.

		ORB-SLAM3 (monocular)			ORB-SLAM3 (stereo)			DSO		
Environment	Sequence	ATE[m]	RPE[m]	duration[s]	ATE[m]	RPE[m]	duration[s]	ATE[m]	RPE[m]	SR
SeaFloor	track0	3.67	**0.13**	34.8 ± 3.17	19.25	0.612	79.38 ± 15.04	-	-	0
	track1	8.78	0.19	64.5 ± 14.9	9.61	0.497	86.52 ± 0.00	2.73	**0.0053**	0.4
	track2	5.18	0.142	40.2 ± 8.9	14.31	0.826	85.46 ± 22.92	17.06	**0.109**	1
SeaFloor Algae	track0	2.99	0.122	31.27 ± 10.4	14.50	0.687	75.26 ± 0.19	3.71	**0.019**	0.4
	track1	1.15	0.134	53.8 ± 17.9	7.02	0.287	83.98 ± 4.08	7.00	**0.048**	1
	track2	9.16	0.144	53.11 ± 9.7	2.99	0.47	64.35 ± 37.47	4.51	**0.044**	0.9
Ocean Floor	track0 dark	5.78	**0.523**	11.3 ± 3.56	3.89	0.880	18.47 ± 0.08	-	-	0
	track0 light	8.37	**0.214**	26.7 ± 5.5	12.42	1.181	91.13 ± 19.19	-	-	0
	track1 light	23.66	**0.286**	64.4 ± 12.8	57.10	1.575	172.8 ± 16.65	-	-	0
Sand Pipe	track0 dark	20.084	**0.184**	83.7 ± 13.7	5.48	1.831	25.72 ± 8.29	-	-	0
	track0 light	6.85	0.0764	76.7 ± 14.5	17.72	5.13	114.64 ± 0.00	10.64	**0.027**	0.7

trajectories are scaled and aligned for the monocular setups using the least-squares alignment in [29]. For each track, ten tests are carried out, from which the median value is depicted in Table IV.

MIMIR-UW comprises very challenging sequences that lead to tracking failure. Consequently, the success rate (SR) is indicated for DSO. The SR is obtained as the number of succeeded tracks completed out of the ten tests carried out in total. Because ORB-SLAM was unable to complete the sequences, the mean duration for each retrieved estimate is provided in this case. DSO provides better performance in those sequences that is able to complete. Higher-textured sequences with higher delentropy values such as SeaFloor and SeaFloor Algae provide lower errors than those with lower texture, like OceanFloor. Although direct SLAM approaches perform better in low-textured areas, that is not the case in dark sequences. Dark sequences provide visibility of nearby objects solely, which produces higher relative motions and, thus, higher parallax, the main cause of failure for direct SLAM. While ORB-SLAM is less affected by parallax, the lack of features in the image makes tracking fail more often. The stereo setup is less prone to tracking loss, but the error is unexpectedly higher. The repeatability of the environment has been identified as the source cause: both monocular and stereo detect incorrect loop closures within the Atlas map, but with a higher frequency in stereo. Consequently, the results show that the current dataset targets the weaknesses and strengths of each SLAM approach to a more significant extent than other existing datasets, opening room for new future lines of development.

B. Segmentation

The semantic segmentation labels provided by MIMIR-UW have been used for training and benchmarking two networks for pipeline segmentation. The networks chosen for that purpose are DeepLabV3 [30] and fully convolutional network (FCN) [31]. FCN replaces the fully connected layers from classification networks with convolutional layers and upsamples the output to recover the input's dimension. This allows fine-tuning of pretrained classification networks for segmentation. DeepLabV3 addresses the spatial information loss caused by convolutional layers by implementing dilated convolutions, which support increasing the capture of semantic context without losing spatial resolution.

Both networks were implemented with ResNet50 as the backbone and using ImageNet pre-trained weights. The network is fine-tuned using an Adam optimizer at a 0.0001 learning rate and cross-entropy loss. The models are trained for a maximum of 50 epochs with early stopping in case of no increase in the validation mean intersection over union (mIoU) over six consecutive epochs. In MIMIR-UW, pipes account for approximately 10% of all pixels in all images. For the training sets, each class is weighted inversely proportional to the number of pixels it occupies in the images. To avoid data redundancy, every fifth image is selected in each track. The segmentation metrics implemented comprise mIoU, and the pixel mean accuracy [32]. They evaluate the overlap between ground-truth and prediction, and the ratio of correctly classified pixels, respectively.

Six experiments have been carried out as depicted in Table V: three with the data contained by the dataset, and three with data from a real pipeline inspection scenario. Considering that generalization ability is one of the open problems in deep learning, we aim to demonstrate the challenges and opportunities for pipeline segmentation that MIMIR-UW proposes when using state-of-the-art segmentation networks. The model in the *SeaFloor* experiment has been trained with tracks from the SeaFloor environment exclusively, and tested against an unseen track from the same environment. Despite of belonging to different tracks, the proximity of the data distribution yields good results for both segmentation networks. Similarly, the *SandPipe* experiment used tracks from the SandPipe environment, trained under the dark sequence and tested in the light one. Despite the similarity in data distribution, the model's performance is inferior to that of *SeaFloor*. One possible reason is how the different lighting conditions affect the object's appearance in the image, yielding a different data distribution for the same pipe model. The last experiment for the simulated data involved all the tracks in the dataset, and it is referred to as *All* in Table V. Here, SeaFloor Algae is only used for testing, while all other environments are used for training and validation. The algae covering the pipes presents an unseen element

TABLE V

EXPERIMENT SETUPS AND RESULTS IN THE TEST SET FOR THE SEGMENTATION NETWORKS TRAINED WITH MIMIR-UW.

			FCN		DeepLabV3	
Experiment name	Data split	(%) sequence{track}{camera}	meanAcc.	mIoU	meanAcc.	mIoU

SeaFloor	Train/Validation
	Test
		(80/20)% SeaFloor{0,1}{cam0}
		100% SeaFloor{2}{cam0}
			0.876	0.785	**0.880**	**0.790**

SandPipe	Train/Validation
	Test
		(80/20)% Sandpipe{dark}{cam0}
		100% Sandpipe{light}{cam0}
			0.643	0.445	**0.726**	**0.522**

All	Train/Validation
	Test
		(80/20)% SandPipe,SeaFloor,OceanFloor{all}{cam0}
		100% SeaFLoor_Algae{all}{cam0}
			0.838	0.700	**0.830**	**0.704**

S2R - SandPipe	Train/Validation Test	(80/20)% Sandpipe{light}{cam2} Real Pipeline Dataset	**0.552**	**0.413**	0.527	0.380
S2R - Augmented SandPipe	Train/Validation Test	(80/20)% Sandpipe{light}{cam2} Real Pipeline Dataset	0.712	0.601	**0.722**	**0.616**
S2R - Augmented SeaFloor	Train/Validation Test	(80/20)% SeaFloor{light}{cam2} Real Pipeline Dataset	0.577	0.440	**0.583**	**0.448**

during training which challenges the model's performance. Nevertheless, the results are close to those in the *SeaFloor* experiment, showing the potential to provide high variability in the training data distribution.

Sim-to-real (S2R) comprises a set of experiments for demonstrating the potential of MIMIR-UW in sim-to-real transfer using the real pipeline inspection dataset as the test set. Figure 4 showcases the differences in the data between the real and the simulated scenarios. Considering that SandPipe provides closer similarity in terms of colour, shape, and disposition for the pipes in the image, the first *S2R* test is carried out with that dataset as shown in Table V. As is to be expected given the difference in the data distribution, the results show much lower performance than that of the experiment performed purely with simulated data. To approximate more closely the appearance of the simulated pipes to the real ones, the next experiment augments the train and validation splits of the SandPipe dataset by performing random flips and random changes in brightness, contrast and saturation. The images are also converted to grayscale and randomly resized. As shown in Table V, increasing the variability of the pipeline appearance significantly improves the models' results. The final *S2R* experiment is carried out with the SeaFloor dataset, applying the same augmentation techniques from the previous experiment. It shows how despite the augmentation strategies, a high difference between the training and test data distribution negatively affects the model's performance.

Figure 5 provides a sample of the results. DeepLabV3's ability to encode spatial information allows it to outperform FCN in almost all scenarios. The experiments highlight the potential of MIMIR-UW to explore open challenges in segmentation, such as generalization across environments, sim-to-real transfer and performance under low-light (and thus low delentropy) environments like SandPipe.

C. Depth estimation

The pertinence of MIMIR-UW for underwater depth estimation is demonstrated by evaluating the performance of two learning-based depth estimation approaches, Monodepth2 [33], and MonoRec [34], on the proposed dataset and the real-world pipeline dataset. Monodepth2 employs a U-net architecture [35] with multi-scale depth estimation. This network takes as input a single RGB image. It predicts a mean normalized inverse depth image by optimizing the per-pixel minimum reprojection error and masking out pixels that do not follow photometric consistency. MonoRec is a framework that consists of a cost volume module that encodes geometric information, a mask module that predicts a photometric inconsistency mask, and a depth module that predicts an inverse depth image. The mask and depth modules both employ neural networks that follow a U-net architecture [35]. This framework takes as input an arbitrary number of RGB images that share a view frustum, the pose transformations between them, and camera intrinsics generated by a structure-from-motion pipeline. MonoRec training consists of four stages, of which only the first stage is considered in this paper for evaluation purposes. The networks are trained on a subset of MIMIR-UW and KITTI [2], to portray the difference between training on underwater and overwater data. The models trained on KITTI are publicly provided by the corresponding authors [33], [34]. In the case of Monodepth, the model trained using monocular and stereo data on an image resolution of 640×192 is used. The minimum and maximum depth used for mean normalization of the inverse depth prediction are set to 0.003m and 80m respectively. The MonoRec model used was the first stage depth bootstrap training, trained on an image resolution of 512×256.

The subset chosen for MIMIR-UW training involves data from the two front cameras of track 0 and track 1 from the SeaFloor environment, consisting of a total of 9760 images. The training procedures for Monodepth and MonoRec first stage are adhered to, barring a few exceptions. Monodepth is trained on an image resolution of 640×192, with a batch size of 32 for 30 epochs, using a pre-trained encoder trained on KITTI. MonoRec is trained on an image resolution of 512×384, with a batch size of 16 for 15 epochs, using pretrained weights from first stage training on KITTI.

The evaluation procedure for both networks is different due to a contrast in inference approach. Monodepth predicts a mean normalized inverse depth image, which is then scaled

TABLE VI
DEPTH ESTIMATION EXPERIMENTS ON MIMIR-UW.

		MonoRec				MonoDepth2			
		KITTI		MIMIR		KITTI		MIMIR	
Environment	Sequence	$SCInv$	*Abs Rel*	$SCInv$	*Abs Rel*	$SCInv$	*Abs Rel*	$SCInv$	*Abs Rel*
SeaFloor	track 0	1.076	2.759	-	-	0.998	0.994	-	-
	track 1	1.054	2.341	-	-	0.786	0.993	-	-
	track 2	1.160	1.792	**0.7200**	**0.3390**	1.094	0.996	**0.7198**	**0.4268**
SeaFloor w. Algae	track 0	1.224	3.071	**1.027**	**0.7697**	1.231	0.999	**0.9612**	**0.7083**
	track 1	**1.027**	2.202	1.112	**0.5147**	1.070	0.997	**0.8889**	**0.5235**
	track 2	1.177	1.929	**1.034**	**0.5657**	1.179	0.995	**0.9705**	**0.6040**
Real	Pipeline	-	1.843	-	**0.2569**	-	0.9455	-	**0.3977**

using the median ratio difference between the predicted inverse depth image and the ground truth inverse depth set it is evaluated on, as specified by the authors [33]. A monocular set of three time-series RGB images, their corresponding ground truth pose transformations, and the camera intrinsics are fed as input to the MonoRec for evaluation. Both approaches are evaluated on the corresponding image resolutions used for training. The evaluation subset chosen from MIMIR-UW involves SeaFloor's track 2, and all tracks from SeaFloor Algae. Additionally, the pseudo ground truth depth for 100 images in the real pipeline dataset is used for evaluation.

The error between ground truth and predicted inverse depth is quantified with the standard metrics scale-invariant error ($SCInv$) and absolute relative distance (*Abs Rel*) [36], yielding the results shown in Table VI. Comparing performance between both networks would be unfair, because Monodepth uses the ground truth depth from evaluation data to influence its result. A comparison between training datasets is considered instead, to validate the importance of MIMIR-UW. The networks trained on MIMIR-UW outperform KITTI training across all of the evaluation data chosen. For Monorec, training on MIMIR-UW yields a lower error with both metrics, on all tracks except track 1 of SeaFloor with algae where the $SCInv$ was slightly higher. This can be attributed to the difficulty of measuring depth for deformable objects like algae, and is confirmed by the difference in $AbsRel$ errors for MIMIR-UW training between SeaFloor and SeaFloor with Algae. The same results can be concluded from training on Monodepth, where MIMIR-UW outperformed KITTI on all metrics. When trained on SeaFloor data, both networks exhibit a lower $AbsRel$ on real data compared to simulation, indicating that the simulation may possess difficulties for the depth estimation task not present in real data. However, a sim-to-real transfer cannot be validated using this evaluation on real data, due to the high level of sparsity of ground truth per depth image (see Fig. 6). Large evaluation metric errors of networks trained on KITTI dataset indicate that neural networks trained on datasets above water cannot generalize well to difficult underwater imaging conditions. Alternately, training on synthetic underwater data in the proposed dataset yields better performance on simulation and real-word data, thus indicating the need for MIMIR-UW.

VI. CONCLUSIONS AND FUTURE WORK

This work presents a multipurpose synthetic underwater dataset for developing computer vision algorithms such as SLAM, segmentation, and depth estimation. We implement delentropy and motion diversity metrics to assess the information contained in the dataset, which are used to compare MIMIR-UW with other datasets. The metrics show that, in comparison to other datasets, the images in MIMIR-UW present more challenging conditions due to dark and blurry images. The low-textured images and high parallax lead to failure and drift in visual SLAM algorithms. Similar conditions also hinder segmentation and depth estimation methods. To the best of our knowledge, MIMIR-UW introduces new challenges for the methods evaluated in this paper, thus opening room for the development of more robust and generalizable methods. Moreover, the potential of MIMIR

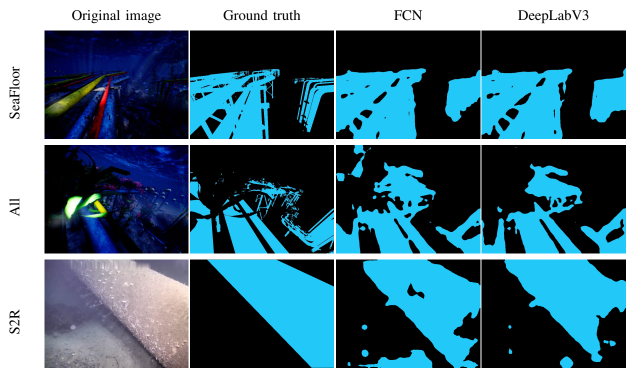


Fig. 5. Sample segmentation results for the models trained in the experiments *SeaFloor*, *All*, and *S2R* trained with augmented SandPipe.

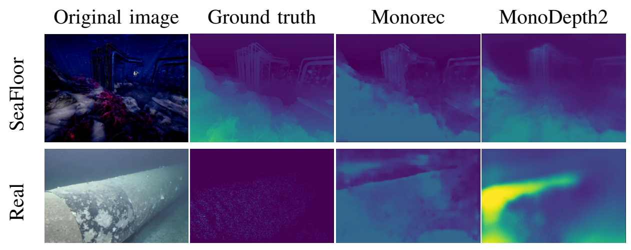


Fig. 6. Sample results on the SeaFloor track and the real pipe for the depth estimation models trained on MIMIR-UW.

for sim-to-real transfer in segmentation and depth estimation has been demonstrated.

Future works include generating more diverse underwater conditions, rotation motions, and annotations for segmentation and optical flow, all within a complete underwater simulation framework with the integration of underwater physics.

Acknowledgment

This paper is written under the project REMARO which has received funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020 under Grant Agreement No 956200. Thanks to EIVA for providing access to real data and allowing sample images' dissemination.

References

[1] J. Agbakwuru, "Oil/gas pipeline leak inspection and repair in underwater poor visibility conditions: challenges and perspectives," *Journal of Environmental Protection*, vol. 2012, 2012.

[2] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[3] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[4] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[5] ETH Zurich Computer Vision Group and Microsoft Mixed Reality & AI Lab Zurich, "The ETH-Microsoft Localization Dataset," https://github.com/cvg/visloc-iccv2021, 2021.

[6] M. Ferrera, V. Creuze, J. Moras, and P. Trouvé-Peloux, "Aqualoc: An underwater dataset for visual–inertial–pressure localization," *The International Journal of Robotics Research*, vol. 38, no. 14, pp. 1549–1559, 2019.

[7] M. Bernardi, B. Hosking, C. Petrioli, B. J. Bett, D. Jones, V. A. Huvenne, R. Marlow, M. Furlong, S. McPhail, and A. Munafò, "Aurora, a multi-sensor dataset for robotic ocean exploration," *The International Journal of Robotics Research*, p. 02783649221078612, 2022.

[8] A. Mallios, E. Vidal, R. Campos, and M. Carreras, "Underwater caves sonar data set," *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1247–1251, 2017.

[9] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4909–4916.

[10] J. Hong, M. Fulton, and J. Sattar, "Trashcan: A semantically-segmented dataset towards visual detection of marine debris," *arXiv preprint arXiv:2007.08097*, 07 2020.

[11] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1769–1776.

[12] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas, and S. S. C Botelho, "Underwater image segmentation in the wild using deep learning," *Journal of the Brazilian Computer Society*, vol. 27, no. 1, pp. 1–14, 2021.

[13] H. I. Ugurlu, X. H. Pham, and E. Kayacan, "Sim-to-real deep reinforcement learning for safe end-to-end planning of aerial robots," *Robotics*, vol. 11, no. 5, p. 109, 2022.

[14] H. X. Pham, A. Sarabakha, M. Odnoshyvkin, and E. Kayacan, "Pencilnet: Zero-shot sim-to-real transfer learning for robust gate perception in autonomous drone racing," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 847–11 854, 2022.

[15] O. Álvarez-Tuñón, A. Jardón, and C. Balaguer, "Generation and processing of simulated underwater images for infrastructure visual inspection with uuvs," *Sensors*, vol. 19, no. 24, p. 5497, 2019.

[16] Y. Song, D. Nakath, M. She, F. Elibol, and K. Köser, "Deep sea robotic imaging simulator," in *International Conference on Pattern Recognition*. Springer, 2021, pp. 375–389.

[17] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "Uuv simulator: A gazebo-based package for underwater intervention and multi-robot simulation," in *OCEANS 2016 MTS/IEEE Monterey*. IEEE, 2016, pp. 1–8.

[18] M. Prats, J. Perez, J. J. Fernández, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *2012 IEEE/RSJ international conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 2577–2582.

[19] "Project dave." https://github.com/Field-Robotics-Lab/dave/wiki., 2023.

[20] E. Potokar, S. Ashford, M. Kaess, and J. Mangelson, "HoloOcean: An underwater robotics simulator," in *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA*, Philadelphia, PA, USA, May 2022.

[21] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*. Springer, 2018, pp. 621–635.

[22] M. Jian, Q. Qi, H. Yu, J. Dong, C. Cui, X. Nie, H. Zhang, Y. Yin, and K.-M. Lam, "The extended marine underwater environment database and baseline evaluations," *Applied Soft Computing*, vol. 80, pp. 425–437, 2019.

[23] K. Panetta, L. Kezebou, V. Oludare, and S. Agaian, "Comprehensive underwater object tracking benchmark dataset and underwater image enhancement with gan," *IEEE Journal of Oceanic Engineering*, 2021.

[24] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[25] K. G. Larkin, "Reflections on shannon information: In search of a natural information-entropy for images," *arXiv preprint arXiv:1609.01117*, 2016.

[26] C. Griwodz, S. Gasparini, L. Calvet, P. Gurdjos, F. Castan, B. Maujean, G. D. Lillo, and Y. Lanthony, "Alicevision Meshroom: An open-source 3D reconstruction pipeline," in *Proceedings of the 12th ACM Multimedia Systems Conference - MMSys '21*. ACM Press, 2021.

[27] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[28] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[29] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[30] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[31] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[32] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[33] C. Godard, O. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11 2019.

[34] F. Wimbauer, N. Yang, L. von Stumberg, N. Zeller, and D. Cremers, "Monorec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[35] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[36] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

Appendix C

Paper - SubPipe: A Submarine Pipeline Inspection Dataset for Segmentation and Visual-inertial Localization

SubPipe: A Submarine Pipeline Inspection Dataset for Segmentation and Visual-inertial Localization

Olaya Álvarez-Tuñón, Luiza Ribeiro Marnet, László Antal, Martin Aubard, Maria Costa and Yury Brodskiy

***Abstract*— This paper presents SubPipe, an underwater dataset for SLAM and image segmentation. SubPipe has been recorded with a lightweight autonomous underwater vehicle (LAUV), operated by OceanScan MST, and carrying a sensor suite including two cameras, an inertial measurement unit (IMU), and a sidescan sonar, among other sensors. The AUV has been deployed in a pipeline inspection environment with a submarine pipe partially covered by sand. The AUV's pose ground truth is estimated from the navigation sensors, and the segmentation labels are manually annotated. State-of-the-art methods on segmentation and SLAM are benchmarked on SubPipe to demonstrate the dataset's challenges and opportunities for leveraging computer vision algorithms. To the authors' knowledge, this is the first annotated underwater dataset providing a real pipeline inspection scenario. The dataset is publicly available online. `https://github.com/remaro-network/SubPipe-dataset`**

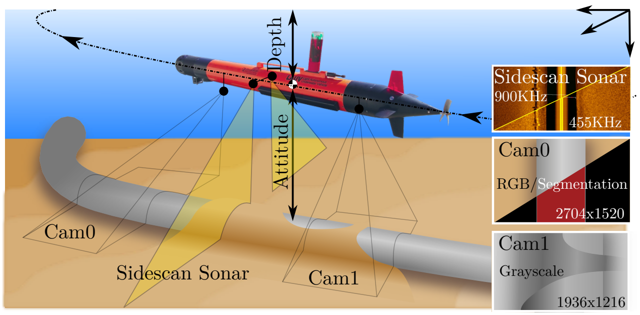


Fig. 1. SubPipe has been recorded during a pipeline inspection mission with OceanScan's LAUV. The recorded data includes two monocular cameras (one monochrome camera and one RGB) with pipe segmentation annotations for the latter one; side-scan sonar images with bounding box annotations of the pipeline for object detection; temperature, altitude, and depth measurements; and the robot's pose, velocity, and acceleration.

I. Introduction

Under the challenging imaging conditions that hinder the performance of computer vision algorithms, underwater vehicles' autonomy has been usually limited to sonar-based methods for localization and detection [1]. Nevertheless, the deep learning paradigm pushes the boundaries of computer vision-based algorithms underwater. Thus, the availability of data becomes the main limiting factor.

While other domains, such as autonomous driving, have long had a wide variety of datasets at their disposal [2], [3], the availability in the underwater domain is limited by the difficulty of robot deployment and ground truth gathering. Offshore structures (such as pipelines) are suitable for autonomous underwater vehicle (AUV) deployment, as they require regular inspections. However, it is often unfeasible to open such data for public use because of privacy and security reasons. Hence, most of the datasets available are recorded in the context of marine life monitoring [4], [5] or archaeological inspection [6]. Moreover, these datasets are often divided between localization, segmentation, or object detection, and rarely provide ground truth for both.

This paper presents SubPipe, a submarine pipeline inspection dataset with ground truth for object detection, image segmentation, and visual-inertial localization. It includes RGB images and side-scan sonar (SSS) images of the seafloor, accelerations captured by an inertial navigation system (INS), as well as the linear velocities estimated by a doppler velocity logger (DVL). Annotations for semantic segmentation are provided for the RGB images from a designated camera, while the side-scan sonar images are annotated for object detection purposes. Other sensor measurements include forward-looking echo sounder, temperature, altitude, pressure, and depth. The outline of the data-gathering process can be seen in Fig. 1.

As shown in Tbl. I, the dataset closest to SubPipe is MIMIR-UW [7], with a similar set of sensors and pipe segmentation labels. However, MIMIR-UW presents a simulated dataset with fewer underwater artifacts, such as scattering and blur. Figure 2 visually compares the imaging conditions between SubPipe and some state-of-the-art underwater datasets.

By releasing SubPipe, we aim to provide the research community with a novel and multimodal underwater dataset, facilitating advancements in underwater computer vision algorithms. To underscore the dataset's significance, we propose a series of experiments that demonstrate the necessity of SubPipe and showcase the novel challenges that state-of-the-art algorithms face in underwater scenarios. Through this contribution, we envision fostering a collaborative effort toward developing robust and versatile underwater computer vision solutions.

The rest of this paper is structured as follows. Section II presents SubPipe, including information about the sensors

O. Álvarez is with Artificial Intelligence in Robotics Laboratory (AiR-Lab), the Department of Electrical and Computer Engineering, Aarhus University, 8000 Aarhus C, Denmark (e-mail: olaya@ece.au.dk). L. Ribeiro and Y. Brodskiy are with EIVA a/s, 8660 Skanderborg, Denmark (e-mails: {lrm,ybr}@eiva.com), L. Antal is with RWTH Aachen University, 52074 Aachen, Germany (e-mail: antal@informatik.rwth-aachen.de), M. Aubard and M. Costa are with OceanScan Marine Systems & Technology, 4450-718 Matosinhos, Portugal (e-mails: {maubard,mariacosta}@oceanscan-mst.com).

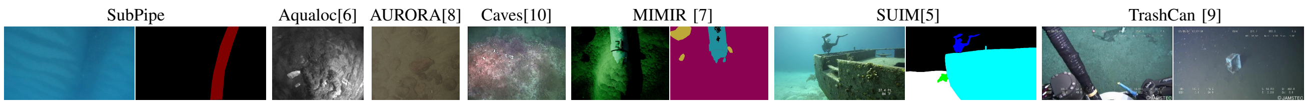


Fig. 2. Sample images from SubPipe and other state-of-the-art underwater datasets. A sample segmentation image is also shown for those datasets that include segmentation labels.

TABLE I

COMPARISON OF STATE-OF-THE-ART UNDERWATER DATASETS. "SEG." AND "DET." INDICATES ANNOTATIONS FOR SEGMENTATION AND OBJECT DETECTION, WHILE "DEPTH" NOTES RANGE-BASED IMAGERY (SUCH AS SIDE-SCAN SONAR AND DEPTH CAMERA).

Dataset	Pose	Seg.	Det.	Camera	Depth	Object labels
Aqualoc [6]	✓	✗	✗	mono	✗	N/A
AURORA [8]	✓	✗	✗	mono	✗	N/A
MIMIR-UW[7]	✓	✓	✗	rig	✓	Marine life; pipes
NAUTEC UWI[4]	✗	✓	✗	mono	✗	Marine life; other
SUIM[5]	✗	✓	✗	mono	✗	Marine life; other
TrashCan[9]	✗	✓	✓	mono	✗	Marine debris
SubPipe (ours)	✓	✓	✓	mono(x2)	✓	Pipe

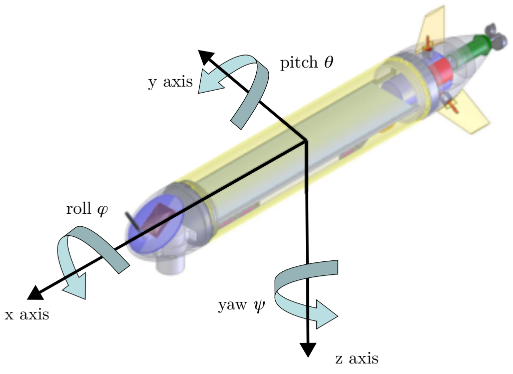


Fig. 3. The 6 degrees of freedom of the LAUV [11].

and techniques used for gathering the dataset. Furthermore, Section III provides insights about the dataset's quality by analyzing different metrics and comparing it with other state-of-the-art datasets. Section IV lists the conducted experiments and their results. Finally, Section V concludes this paper.

II. THE DATASET

SubPipe has been recorded during a survey mission performed by OceanScan-MST[1], near an underwater pipeline with about 1 km length in Porto, Portugal. The data recorded during the mission corresponds to the data measured by the onboard sensors and the estimated state of the robot as inferred from the sensors.

The onboard sensors are: two downward-looking cameras with 3-channel 2704×1520 and 1-channel 1936×1216 images, respectively; a Klein 3500 side-scan sonar, providing sonar images recorded at 900 KHz and 455 KHz, with a range of 30m per transducers, resulting in monochrome waterfall images with sizes 5000×500 and 2500×500,

[1]OceanScan's home page: `https://www.oceanscan-mst.com/`

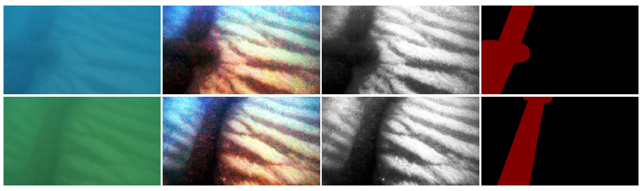


Fig. 4. Segmentation mask generation process illustrated with two examples. Each row presents a unique example. From left to right: the original raw image, the image after histogram equalization of RGB channels, the original image after converting to grayscale and applying histogram equalization, and the final segmentation mask. The equalization process enhances contrast, aiding in the precise delineation of the pipeline, including the pipe clamp, which is annotated as part of the pipeline.

respectively. Further details on the dataset, such as the exact number of frames and the intrinsic and extrinsic parameters of the cameras, the side-scan sonar, and other sensors, can be found in the online repository of SubPipe.

The estimated state corresponds to the 6 DOF pose of the robot (see Fig. 3) as inferred from the INS and DVL. The three translational measurements, surge x, sway y, and heave z, are recorded in meters and are estimated by a Kalman filter considering the last recorded GPS coordinates and measured velocities. On the other hand, the rotational measurements roll ϕ, pitch θ, and yaw ψ are provided in radians and measured by a gyroscope. Besides, SubPipe provides other measurements of i. the AUV (e.g., depth, altitude, velocities, angular velocities, accelerations, rpm, forward distance measured from the nose of the AUV) and ii. the environment (water velocity, temperature, pressure).

SubPipe provides five video sequences, ranging between approximately 7 and 9 minutes. The high-resolution camera has been recorded at 240 Hz, and the low-resolution one at 4 Hz. All the SSS waterfall images have been extracted and manually annotated using the COCO format for object detection. This process created a single class named *Pipeline*. The sonar images were resized to a dimension of 640×640 pixels for compatibility with computer vision algorithms.

An overview of the data contained in each of the sequences (or chunks) is depicted in Tbl. II. Given the large dataset size, a smaller subsample of SubPipe referred to as SubPipeMini, is provided. SubPipeMini contains approximately 20% of the complete sequence data. Segmentation labels for the class 'pipeline' were manually annotated for every 25 frames of the 16170 high-resolution images of SubPipeMini, resulting in 647 labeled images. The annotation process was conducted using the LabelMe tool [12]. The repetitive nature of the underwater environment led to the decision to label the frames selectively, reducing redundancy in the labeled dataset. The dataset's pipeline visibility is significantly reduced due to considerable blurring. To facilitate the labeling process, the

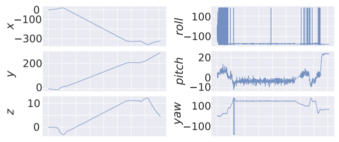
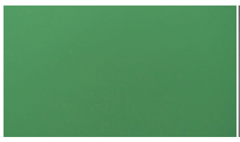
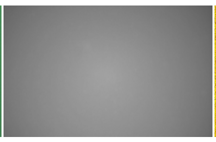
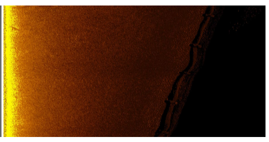
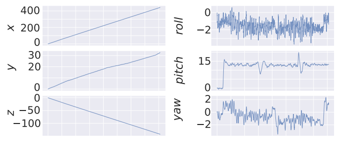
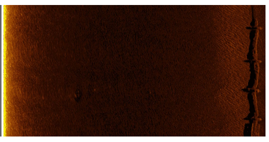
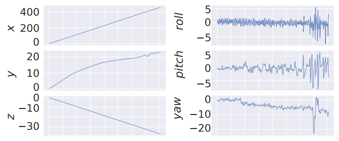
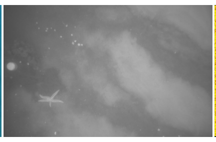
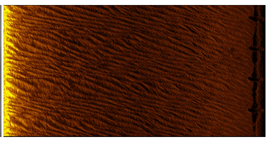
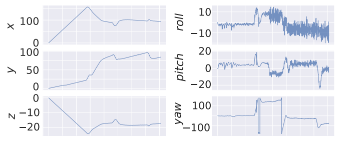
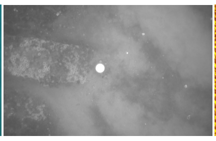
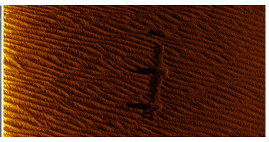
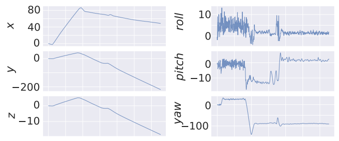
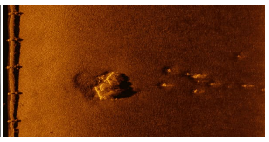
Chunk		Trajectory [m] / [deg]	Cam0	Cam1	Side-scan
0	Time: 9m 0.54s Length: 571.7m #Poses: 16199				
1	Time 9m 0.54s Length: 466.8m #Poses: 16199				
2	Time: 8m 59.54s Length: 463.4m #Poses: 16169				
3	Time: 8m 59.54s Length: 372.7m #Poses: 16169				
4	Time: 7m 15.73s Length: 374.1m #Poses: 13057				

TABLE II

OVERVIEW OF THE DATA FOR EACH SUBPIPE CHUNK, INCLUDING SAMPLE IMAGES FROM THE CAMERAS AND THE SIDE-SCAN SONAR.

images' contrast was enhanced using histogram equalization, as depicted in Fig. 4. Note that these preprocessed images, though crucial for label generation, are not part of the released dataset.

Since the high-resolution camera was not synchronized with the rest of the vehicle's sensors, no exact sensor measurements were available for the timestamps of the extracted video frames. To solve this issue, we used the recorded sensor measurements, and by applying linear interpolation, we calculated the estimated sensor values corresponding to the timestamp of each frame, formally:

$$\nu^*(t) = \frac{\nu_1 - \nu_0}{t_1 - t_0} * (t - t_0) + \nu_0, \tag{1}$$

where $\langle t_0, \nu_0 \rangle$ and $\langle t_1, \nu_1 \rangle$, represent the (closest in time) previous and following timestamp and measurement values, correspondingly. At the same time, t is the timestamp for which we calculate the estimate, and finally, ν^* is the estimated value. This interpolation is applied for all sensor measurements, providing the estimated values for the timestamps corresponding to the extracted camera images.

III. DATASET METRICS

The information contained by the dataset is quantitatively evaluated and compared with existing state-of-the-art datasets using the metrics proposed in [7] and [14]: delentropy and motion diversity.

The delentropy (or image entropy) H is calculated for each image in the dataset to measure the amount of information it contains. It is based on Shannon's joint entropy formulation, taking the image's gradient in the x and y axes as the joint probability density function. A uniform distribution of the joint probability density function for the gradient yields maximum values for H. The delentropy is maximized under the presence of nondistinctive features and minimized (zeroed) for zero-gradient images containing no features. Ideal conditions are then represented by intermediate values that indicate the presence of distinctive features.

The motion diversity metric σ is a function of the principal components of the sequence of relative motions. The motion diversity metric converges to one with evenly distributed motion sequences, and to zero under the presence of motion in only one axis.

Figure 5 depicts the result of deploying the proposed metrics on SubPipe and the datasets for visual localization Aqualoc [6], MIMIR [7], EuRoC [15], KITTI [13], and TartanAir [14]. This deployment is driven by the specific design of these metrics for localization datasets. Aqualoc and MIMIR are real and synthetic underwater datasets, respectively. TartanAir includes a high diversity of synthetic tracks, including underwater environments. EuRoC and

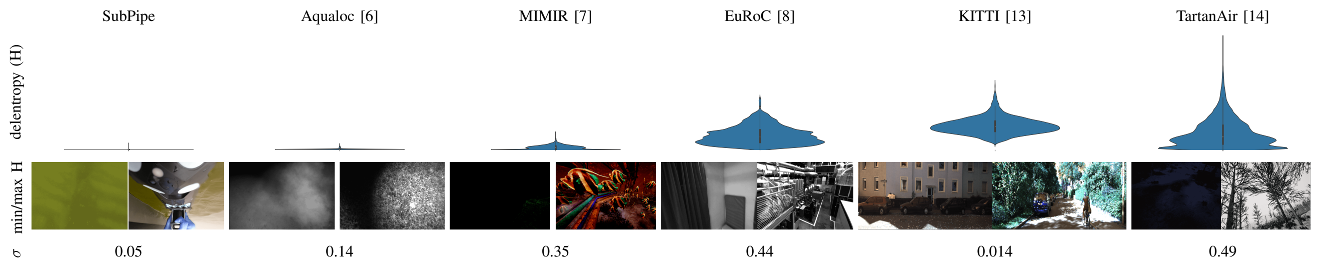


Fig. 5. Dataset metrics. The top section presents the distribution of delentropy values across various datasets, accompanied by representative images that yield the minimum and maximum delentropy. On the bottom are the motion diversity metric results. Lower delentropy and motion diversity values indicate a high degree of uniformity in image content and limited motion variety across the six degrees of freedom. This phenomenon is particularly evident in SubPipe, since pipeline inspection missions inherently limit motion and imaging variability. In contrast, datasets such as EuRoC, featuring a drone navigating freely in six degrees of freedom within an indoor environment, exhibit significantly more diversity in both imaging and motion.

KITTI are above-water datasets recorded with a drone and a car, respectively. Noticeably, above-water datasets retrieve much more information-rich images than underwater ones. Underwater datasets have a higher presence of images with low sharpness due to textureless or blurred areas caused by the seabed's uniformity and light scattering, among other phenomena. Nevertheless, motion blur and low illumination frames can also retrieve low entropy values above water, as seen in EuRoC. MIMIR presents richer images among the underwater datasets, as is to be expected in simulated images. SubPipe's images present delentropy values very close to zero in almost all images, originating from the uniformity of the background and the lowly illuminated sequences. Regarding motion diversity, drone sequences like the ones in EuRoC and TartanAir present higher diversity. SubPipe, however, presents a low diversity, as is expected for a pipeline inspection mission: the robot follows a mainly straight path, only altered by the seabed's and the pipe's shape, which the robot must follow. In that sense, the diversity of the motion is similar to a car's motion, with the motion, in this case, constrained by the degrees of freedom of the car. That is evidenced by the similarity between SubPipe and KITTI's motion diversity metrics.

IV. Experimental evaluation

This section proposes a set of experimental evaluations for SubPipe, consisting of leveraging state-of-the-art algorithms for RGB image segmentation, visual SLAM, and object detection in side-scan sonar images.

A. Visual SLAM

The proposed state-of-the-art SLAM algorithms are the geometry-based algorithms ORB-SLAM3 [16] and DSO [17], and the learning-based method TartanVO [18].

ORB-SLAM3 integrates an indirect-based approach that relies on ORB descriptors that are efficiently tracked across frames following a constant velocity model. Meanwhile, an Atlas map is created, in which, under tracking loss, the current map is stored, and a new map is created. These maps are merged into one if a loop is detected. Indirect methods like ORB-SLAM3 rely on descriptor matching and thus are prone to fail in low-textured areas such as the ones present in SubPipe. Conversely, direct methods like the proposed DSO compare pixel intensities across image frames, therefore using more information from the image. These methods are more robust to low-textured areas but more sensitive to large baselines. DSO, in particular, presents a sparse approach that tracks all those pixels in the image with a high gradient. The learning-based method TartanVO, as described in [18], leverages PWC-Net [19] for computing optical flow and employs a tailored version of ResNet50 [20] for pose estimation, featuring dual output branches dedicated to rotation and translation respectively. Furthermore, it integrates the intrinsic camera parameters via an intermediary layer that bridges the optical flow computation and pose estimation components, enhancing its adaptability to various camera parameters. The loss function proposed is a combination of optical flow and camera motion.

Both geometry-based algorithms fail to track the trajectory under SubPipe's very challenging imaging conditions. The lack of features and gradients in the image, as evidenced by the delentropy metric in Section III, is the source of failure. ORB-SLAM3 can track some pipe sections, as depicted in Fig. 6. The pipe clamps provide a higher density of distinctive features that the algorithm can track. However, the track is lost as soon as those areas are out of the cameras' field of view. Similarly, the uniformity of the pixel gradients does not provide DSO with enough points to track across frames, as shown in Fig. 7.

The lack of information on SubPipe's images challenges the performance of geometry-based algorithms. Under these conditions, there is a potential for visual localization algorithms to benefit from the higher-level representations of the image that learning-based approaches can achieve [21]. Thus, the next set of proposed experiments takes TartanVO as the baseline architecture for leveraging learning-based visual odometry architectures. The experiments involve leveraging TartanVO's original model on SubPipe first and then a fine-tuned model. The last layers of the optical flow module and the pose regressor were fine-tuned with SubPipe's chunk 2, which was afterward tested in chunk 3. Given that SubPipe does not provide optical flow ground truth, the loss function corresponds to the chordal loss for the SE(3) pose introduced in [22]. It was fine-tuned for 15 epochs and with an Adam optimizer with an initial learning rate of 10^{-4} and weight

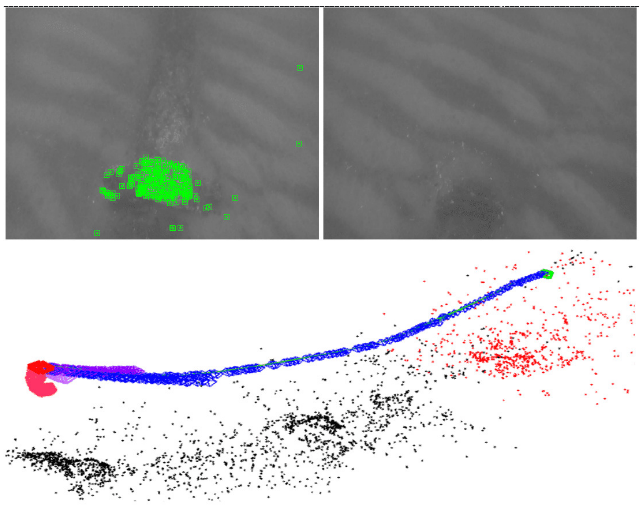


Fig. 6. ORB-SLAM3's performance on SubPipe. The pipe clamps (top-left) provide enough features to track the camera's motion; however, the track is lost once the clamps and the pipe are out of sight (top-right).

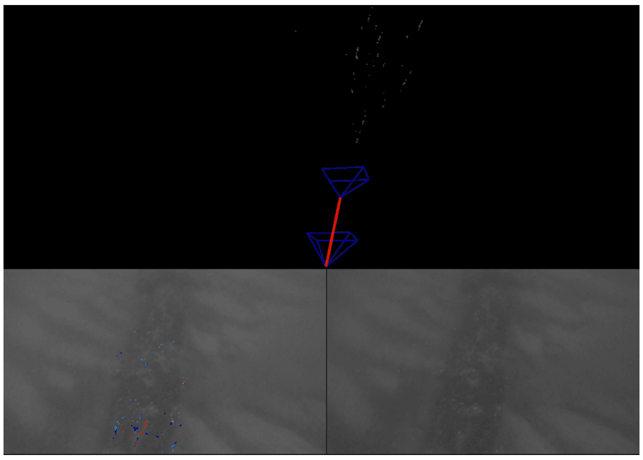


Fig. 7. DSO's performance on SubPipe. The low gradients do not allow the detection of enough features (the colored pixels in the bottom-left image) to be tracked after the algorithm's initialization.

TABLE III

Visual Odometry results on SubPipe's chunk 3.

TartanVO (original model)		TartanVO (fine tuned w. SubPipe)	
ATE[m]	RPE[m]	ATE[m]	RPE[m]
182.3	0.4	66.5	0.02

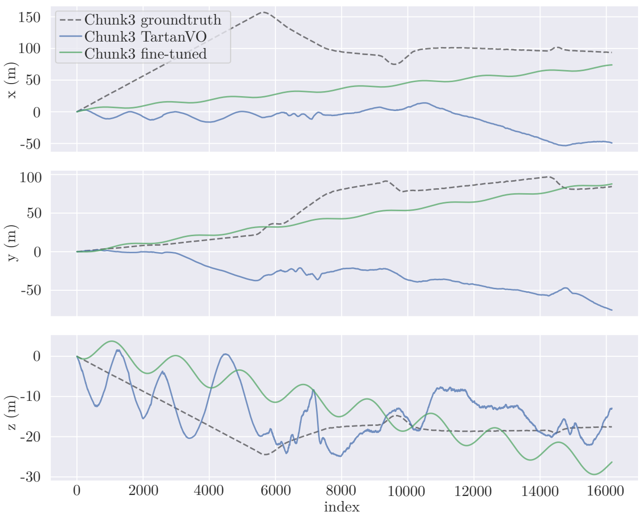


Fig. 8. Results of applying TartanVO's models in SubPipe's Chunk 3. In blue, the original model provided by the authors. In green, the original model fine-tuned with SubPipe's Chunk 2.

decay factor of 10^{-3}. Table III shows the absolute position error (APE) and the relative position error (RPE) [23] metrics resulting from comparing the ground-truth with the results inferred from both models. While the error is considerably lower in the fine-tuned model than in the original model, Fig. 8 evidences that such low error indicates that the model overfitted to the data. This is caused by the uniformity of the images and the trajectory, as evidenced by the dataset metrics in Section 5. Therefore, additional datasets in the fine-tuning process must be included, introducing a broader data distribution.

B. RGB Image Segmentation

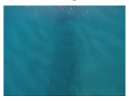
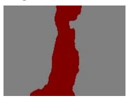
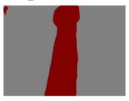
The models chosen for the RGB image segmentation experiments were SegFormer [24] and DeepLabV3 [25]. SegFormer is a visual transformer that uses transformer blocks in the encoder and uses the encoder's multiscale output features as inputs for a multilayer perceptron decoder. Using multiscale features in the decoder helps combine local and global information to achieve better results. This model was implemented in Pytorch and trained from scratch.[2] DeepLabV3 is a largely used convolutional neural network (CNN) for image segmentation that uses dilated convolution, which gives a better field of view for the filters, allowing for better detection of objects in different scales. This second model used the official PyTorch implementation with weights pre-trained on a subset of the COCO dataset.

Both models were trained with cross-entropy loss, using the Adam optimizer with an initial learning rate of 10^{-4}.

[2]Reference implementation: `https://github.com/FrancescoSaverioZuppichini/SegFormer`

TABLE IV

Sample results (top) and mIoU (bottom) for segmentation.

Image	Groundtruth	SegFormer [24]	DeepLabV3 [25]
			
	Background [%]	91.76	91.21
Results	Pipeline [%]	66.42	60.12
	mIoU [%]	79.09	75.66

The models were trained using the 647 annotated frames. The first 60% labeled images from SubPipeMini were used for training, the next 20% for validation, and the last 20% for testing. A suite of augmentation techniques was employed to mitigate overfitting, including flipping, rotation, cropping, and perturbations in RGB channels, saturation, and hue.

Even though SegFormer was trained from scratch, it still achieved good results, recognizing the pipeline relatively well. It indicates that the data has enough quantity and quality to train a deep-learning model. At the same time, even though the pipeline is a straight line, with an easy shape to be segmented, the intersection over union (IoU) for this class has room for improvement for both models, highlighting the difficulties of processing underwater images. Since pipeline tracking and inspection is a critical task, ideally, the IoU for the pipeline should be improved. Perhaps pre-processing the images, using techniques to, e.g., increase the contrast, could improve the results.

C. *Object Detection in Side-Scan Sonar images*

SubPipe provides SSS images featuring underwater pipelines. To evaluate the dataset's validity and quality, the YOLOX object detection model has been employed in two sizes: YOLOX-S with 9 million parameters and YOLOX-Nano with 0.9 million parameters [26]. The models were trained on high-frequency data from timestamp `1693573388` to `1693574657`, providing 15,404 images divided into 70% for training, 15% for testing, and 15% for validation. Both models used pre-trained weights from the COCO dataset and trained during 300 epochs. The validation results on the training data are presented in the first column of Tbl. V, indicating a 98% accuracy for YOLOX-S and 96% for YOLOX-Nano.

To further ensure the dataset's validity, a separate SSS mission from a different survey provided an additional validation set, from which 7,900 images were extracted. The second validation's results are in the second column of Tbl. V.

A comparative analysis of both validation datasets revealed that neither model adequately detected objects in the second, validation dataset. Figure 9 includes two samples from each dataset, illustrating that despite being collected from the same location, the datasets possess distinct characteristics, such as sand, pipeline shadow, and pipeline size variations. These characteristics implicitly offer insights into the models' learning patterns. Consequently, the experimental results suggest that even with a sufficient dataset size and high training metric values, environmental factors such as the sand's properties and the vehicle's altitude (affecting the pipeline shadow) must be considered for effective pipeline detection.

TABLE V
OBJECT DETECTION RESULTS.

Model	AP_{50-95} Training	AP_{50-95} Validation
YOLOX-S	98.1%	14.9%
YOLOX-Nano	96.3%	10.06%

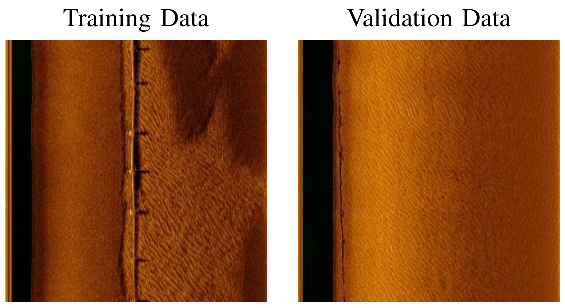


Fig. 9. Comparison between training and validation datasets. The sample on the left illustrates the pipeline and its environment as represented in the training set, while the sample on the right depicts the validation dataset. The pipeline is next to the nadir gap in the second image.

V. CONCLUSION AND FUTURE WORK

SubPipe is an underwater pipeline dataset for visual-inertial SLAM, object detection and segmentation tasks. It presents challenging imaging conditions characteristic of underwater environments. This paper proposes a set of experiments consisting of deploying state-of-the-art algorithms to benchmark those challenges. These experiments underscore the challenges and insights in deploying visual SLAM, image segmentation, and object detection in the context of submarine pipeline inspection.

Geometry-based visual SLAM algorithms struggle with SubPipe's low-textured areas, leading to track loss or inadequate feature tracking, whereas the learning-based TartanVO shows promise in navigating these conditions. For RGB image segmentation, both SegFormer and DeepLabV3 show promising results despite the complexities of underwater imagery. Object detection experiments on SSS data demonstrate the influence of recording conditions on the performance of the YOLOX model. These experiments indicate that the model's accuracy may be compromised even when using the same SSS system, frequency, and pipeline, highlighting the inherent challenges in applying object detection to SSS images.

Collectively, these findings accentuate the promise of learning-based algorithms for underwater computer vision and the need for wider availability of data recorded under such conditions that allow more general models. Consequently, the public release of this dataset serves as a critical contribution to addressing the scarcity of publicly available underwater datasets for computer vision.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 956200. For further information, please visit `https://remaro.eu`.

Acknowledgements

This project has received funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020 under the Grant Agreement No 956200.

References

[1] M. Aubard, A. Madureira, L. Madureira, and J. Pinto, "Real-time automatic wall detection and localization based on side scan sonar images," in *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*, pp. 1–6, IEEE, 2022.

[2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3213–3223, June 2016.

[3] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009. Video-based Object and Event Analysis.

[4] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas, and S. S. C. Botelho, "Underwater image segmentation in the wild using deep learning," *Journal of the Brazilian Computer Society*, vol. 27, pp. 1–14, Oct 2021.

[5] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1769–1776, IEEE, 2020.

[6] M. Ferrera, V. Creuze, J. Moras, and P. Trouvé-Peloux, "Aqualoc: An underwater dataset for visual-–inertial-–pressure localization," *The International Journal of Robotics Research*, vol. 38, no. 14, pp. 1549–1559, 2019.

[7] O. Álvarez Tuñón, H. Kanner, L. R. Marnet, H. X. Pham, J. le Fevre Sejersen, Y. Brodskiy, and E. Kayacan, "MIMIR-UW: A multipurpose synthetic dataset for underwater navigation and inspection," *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6141–6148, 2023.

[8] M. Bernardi, B. Hosking, C. Petrioli, B. J. Bett, D. Jones, V. A. Huvenne, R. Marlow, M. Furlong, S. McPhail, and A. Munafò, "Aurora, a multi-sensor dataset for robotic ocean exploration," *The International Journal of Robotics Research*, vol. 41, no. 5, pp. 461–469, 2022.

[9] J. Hong, M. Fulton, and J. Sattar, "Trashcan: A semantically-segmented dataset towards visual detection of marine debris," *CoRR*, vol. abs/2007.08097, 07 2020.

[10] A. Mallios, E. Vidal, R. Campos, and M. Carreras, "Underwater caves sonar data set," *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1247–1251, 2017.

[11] U. do Porto Faculdade de Engenharia LSTS, "Imc navigation messages - estimated state." https://lsts.pt/docs/imc/master/Navigation.html#estimated-state, 2024-01-24.

[12] K. Wada, "Labelme: Image Polygonal Annotation with Python." https://github.com/wkentaro/labelme, 2016.

[13] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361, IEEE, 2012.

[14] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4909–4916, IEEE, 2020.

[15] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[16] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[17] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[18] W. Wang, Y. Hu, and S. Scherer, "Tartanvo: A generalizable learning-based vo," in *Proceedings of the 2020 Conference on Robot Learning* (J. Kober, F. Ramos, and C. Tomlin, eds.), vol. 155 of *Proceedings of Machine Learning Research*, pp. 1761–1772, PMLR, 16–18 Nov 2021.

[19] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8934–8943, 2018.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learningfor image recognition," *CoRR, abs/1512*, vol. 3385, p. 2, 2015.

[21] O. Álvarez-Tuñón, Y. Brodskiy, and E. Kayacan, "Monocular visual simultaneous localization and mapping:(r) evolution from geometry to deep learning-based pipelines," *IEEE Transactions on Artificial Intelligence*, 2023.

[22] O. Alvarez-Tunon, Y. Brodskiy, and E. Kayacan, "Loss it right: Euclidean and riemannian metrics in learning-based visual odometry," in *ISR Europe 2023; 56th International Symposium on Robotics*, pp. 107–111, VDE, 2023.

[23] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, IEEE, 2012.

[24] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 12077–12090, Curran Associates, Inc., 2021.

[25] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017.

[26] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: exceeding YOLO series in 2021," *CoRR*, vol. abs/2107.08430, 2021.

Appendix D

Paper - Bridging the Sim-to-Real GAP for Underwater Image Segmentation

Bridging the Sim-to-Real GAP for Underwater Image Segmentation

Luiza Ribeiro Marnet[†,‡], Stella Grasshof[‡], Yury Brodskiy[†], and Andrzej Wąsowski[‡]
[†]EIVA a/s, Denmark, [‡]Computer Science Department, IT University of Copenhagen, Denmark

***Abstract*— Labeling images for every new task or data pattern a model needs to learn is a significant time bottleneck in real-world applications. Moreover, acquiring the necessary data for training the models can be challenging. Ideally, one would train the models with simulated images and adapt them for the desired real tasks using the least possible amount of data. Active learning can be used to solve this problem with minimal effort. In this work, we train SegFormer for pipeline segmentation with synthetic images from an underwater simulated environment and fine-tune the model with real underwater pipeline images recorded in a marina. The evaluation shows that selecting real data with active learning for fine-tuning the model gives better results than randomly selecting the images. As part of the work, we release the dataset recorded in the marina, MarinaPipe, which will be publicly available.**

***Index Terms*— active learning, computer vision, sim-to-real, underwater image segmentation**

I. Introduction

Training deep learning models requires good-quality datasets. Acquiring and labeling data is costly and time-consuming, making synthetic data an attractive option. Although modern simulators are highly advanced and close to real scenarios, a gap still exists between their patterns.

One way to address this gap is to pre-train a model using synthetic data and then fine-tune it with real data. Yet, this means acquiring and annotating the dataset, which typically requires hours of labeling by specialists. Nevertheless, the datasets usually contain many repetitive patterns that pre-trained models already recognize. Active learning can be applied to select the minimal subset of samples that needs to be used for fine-tuning, based on the model's lack of knowledge [1].

In this paper, we study this sim-to-real gap and how to overcome it in the underwater vision domain, cf. Fig. 1. Real underwater images pose many challenges, including non-uniform illumination, low contrast, color degradation, and motion blur [2], [3], [4], [5]. These are properties that are challenging to mimic realistically in synthetic images, making this study relevant. To the best of our knowledge, this is the first paper studying the sim-to-real gap for RGB underwater images using active learning. Our contributions include:

- The use of active learning to fine-tune a model trained with synthetic data using underwater real data;
- An evaluation of the visual transformer SegFormer with the active learning technique;
- An underwater pipeline dataset, MarinaPipe, publicly released.

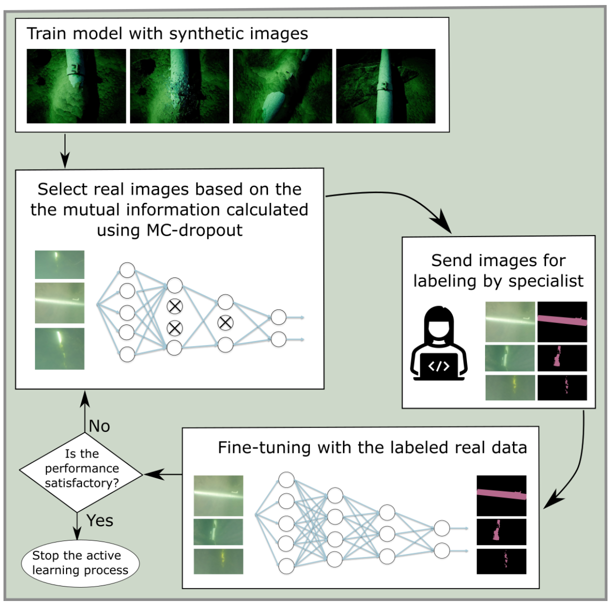

Train model with synthetic images
Select real images based on the the mutual information calculated using MC-dropout
Send images for labeling by specialist
No
Is the performance satisfactory?
Yes
Stop the active learning process
Fine-tuning with the labeled real data

Fig. 1: Sim-to-Real with Active Learning.

Our experiments show that active learning gives better results than random selection for overcoming the sim-to-real gap. Moreover, the model pre-trained with synthetic data and fine-tuned with real data presented better performance than the model trained only with real data in almost all image sequences tested. It is worth using active learning when having a limited budget for labeling images, since it ensures the best outcomes.

II. Background and Related Work

Active Learning methods train machine learning models in an iterative way, beginning with training the model with an initial subset of samples, typically randomly selected, and then retraining it with strategically chosen new samples. To cleverly select these new samples, the uncertainties of the predictions made by the trained model are computed. These uncertainties reflect the knowledge, or lack thereof, that the model has about the input samples. Samples with high uncertainty are those that the model has less knowledge about and, therefore, are more beneficial for using while retraining the model, contrary to the samples with low uncertainty, about which the model already possesses enough knowledge.

Active learning has been extensively studied with the goal of training models with the minimal amount of data necessary to achieve results comparable to the models trained with the entire dataset. In medical images, for instance, the softmax confidence was used as a measure of uncertainty when segmenting pulmonary nodules [6] and membrane images [7]. However, softmax can be overconfident and present high confidence values during the inference phase for samples that are out-of-distribution in relation to the training dataset [8]. Other studies using medical images applied Monte Carlo Dropout (MC-Dropout) [9] for calculating metrics such as max-entropy and Bayesian active learning by disagreement (BALD) as a measure of uncertainty [10], [11]. In the context of autonomous driving cars, which is closer to underwater inspection, entropy-based metrics were used for selecting images for training segmentation models [12], [13], [14].

Since active learning methods retrain the models with the least amount of data by detecting the samples that the models do not have sufficient knowledge about, they can be applied to help overcome the sim-to-real gap with minimal effort [1]. In this work, we train an underwater pipeline segmentation model using the synthetic dataset MIMIR [15]. Subsequently, we utilize active learning to select the most relevant images in a real underwater dataset for fine-tuning the previously trained model, therefore adapting it to the specific chosen dataset. To calculate the uncertainty used for querying new images with active learning, we employ MC-Dropout, a method largely studied in the deep learning community for accessing the epistemic uncertainty, which arises from the model's lack of knowledge [16], [17]. This method consists of allowing the dropout layers [18] during the inference time. Dropout layers temporarily remove neurons in a specific layer with a chosen probability. It means that when these layers are allowed, the same input can have different outputs if forward passed through the model more than once. These layers are originally used during training to prevent overfitting [18], but can be used for accessing the epistemic uncertainty during inference [9]. If the outputs for several forward passes of the same input are similar, it means that the model has enough knowledge about that input pattern; if the outputs are very different from each other, it indicates a lack of knowledge.

III. Methodology

In this section, we present the methodology and details for our experiments with pipeline image segmentation: Sec. III-A presents the pre-training phase using the synthetic dataset MIMIR [15], Sec. III-B the fine-tuning phase for overcoming the sim-to-real gap, Sec. III-C the details about the model structure and the training, and Sec. III-D the datasets used.

A. Pre-Training with Synthetic Data

The first step of the experiments was to train a segmentation model on the synthetic dataset MIMIR [15]. MIMIR has several environments, and we used the one called SandPipe, which contains a single pipeline on the ocean floor. The images were captured by a camera placed at the bottom of the autonomous underwater vehicle (AUV) in the simulated environment, similar to the position of the camera that collected the real dataset.

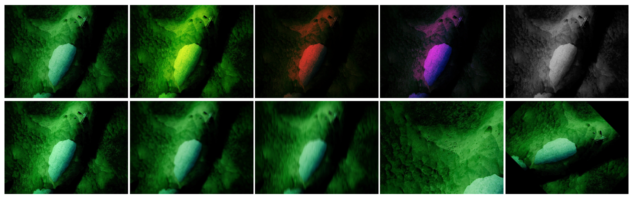


Fig. 2: Examples of augmentations applied to MIMIR. Top: The most left image is the original; the next three are examples of RGB channels perturbation; and the most right is a conversion to grayscale. Bottom (from left-to-right): Value and saturation perturbation; Gaussian blur addition; motion blur addition; resize with cropping; and rotation. In the images with blur addition, notice how the object contours get weaker and how the screw in the pipeline joint "disappears" (Best viewed online in color with zoom-in.)

We chose to use a visual transformer SegFormer [19] for segmentation. We modified it to include dropout layers. The final goal was to use the trained model with real underwater images. To reduce the overfitting to MIMIR [15], smoothing the transition from simulation to reality, several randomized augmentations were performed, Fig. 2:

- Perturbation of the RGB channels and the value and saturation in the HSV color space;
- Resizing, cropping and flipping;
- Conversion to grayscale;
- Addition of motion and Gaussian blur.

B. Fine-Tuning with Real Data

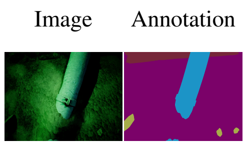
After training the model with MIMIR [15], the active learning method was applied to select the most relevant images from the real underwater dataset for fine-tuning the model. The mean epistemic uncertainty over all pixels of each image was used as the acquisition function for querying new images with the active learning method. For calculating the mean value for each image, the epistemic uncertainty of each pixel was first calculated. This work uses the mutual information, $\mathcal{I}$, calculated with MC-Dropout, as the epistemic uncertainty. The MC-Dropout is applied during the inference phase and consists of forward passing each input sample T times. For a dataset with C classes, at each forward pass t, the model generates for each pixel a softmax output equal to $p_t = (p_{1t}, ..., p_{Ct})$. Using the outputs p_t, the mutual information of each pixel is:

$$\mathcal{I} = \mathcal{H} + \frac{1}{T\log_2(C)} \sum_{c=1}^{C} \sum_{t=1}^{T} p_{ct} \log_2(p_{ct}), \qquad (1)$$

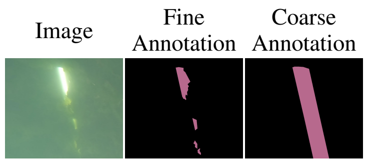
where $\mathcal{H}$ is the entropy, calculated as:

$$\mathcal{H} = \frac{-1}{\log_2(C)} \sum_{c=1}^{C} p_c^* \log_2(p_c^*), \qquad (2)$$

where $p^* = (p_1^*, ..., p_C^*)$ is the average of the predictions p_t over the T forward passes. Equation (1) and Eq. (2) where divided by $log_2(C)$ to normalize the entropy between 0 and 1.



(a) MIMIR



(b) MarinaPipe

Fig. 3: MIMIR and MarinaPipe samples.

Finally, the image uncertainty, EU_{img}, was calculated as:

$$EU_{\text{img}} = \frac{1}{N}\sum_{i=1}^{N} EU_i, \quad (3)$$

where EU_i is the mutual information $\mathcal{I}$ defined in Eq. (1) for the pixel i of an image with N pixels. Images with high EU_{img}, above a chosen threshold TR, were selected for fine-tuning the SegFormer model that was pre-trained with MIMIR [15]. The threshold TR was defined as:

$$\text{TR} = \overline{EU}_{\text{img}} + S\sigma, \quad (4)$$

where $\overline{EU}_{\text{img}}$ and σ are the mean and standard deviation of the values of EU_{img} computed for the MIMIR images using the pre-trained SegFormer. The variable S is a scalar value defined by the user. Smaller values of S ensure a more rigid tolerance with the uncertainty but mean querying more images for being labeled. Notice that two values of TR are calculated, one for the training and the other for the validation dataset. More details on how to query new images with the active learning framework are in our previous work [20].

C. Model Structure and Training Details

The segmentation model used in this study was the visual transformer SegFormer [19] implemented in PyTorch.[1] We modified the structure for including dropout layers in the encoder.

The model was pre-trained from scratch, for 600 epochs, using cross-entropy loss, Adam optimizer, and an initial learning rate of 10^{-4}. During the fine-tuning phase with real data, the decoder and encoder of SegFormer were frozen, and only the head of the model was allowed to train. At this phase, the model was trained for 100 epochs, using cross-entropy loss, Adam optimizer, and an initial learning rate of 10^{-5}. During both pre-training and fine-tuning, the classes used were background and pipeline.

D. Datasets

Two datasets were used in this paper, cf. Fig. 3.

1) MIMIR: a synthetic multipurpose dataset originating in a prior study [15], tailored for pipeline tracking, created in a simulation environment with automatic pixel-wise labeling for many classes, including pipeline. MIMIR has several environments, with SandPipe being one of them. SandPipe has images of a single pipeline, positioned on the ocean floor. This environment has images recorded from three cameras in a simulated AUV. In this study, we use the images from SandPipe recorded from the bottom of the AUV facing down the ocean floor. These images were selected because the images on the MarinaPipe dataset are visually closer to them than to the images recorded by the other cameras and on the other environments of MIMIR.

2) MarinaPipe: a real underwater dataset, recorded in a marina close to the north of Portugal, by our partner OceanScan-MST. The dataset contains pieces of pipes placed on the marina floor, filmed using a GoPro camera attached to a lightweight autonomous underwater vehicle (LAUV). Seven videos were recorded at 240 frames per second (FPS), from which we extracted five frames per second to create the dataset. In some videos, the pipes are partly occluded by algae. For performing the experiments, which the results are described in Sec. IV-B, 10% of the frames of each video were labeled for the task of pipeline segmentation. Table I provides an overview of the MarinaPipe dataset. This dataset was originally idealized for training a model that would later be tested for tracking long pipelines. Because of this, the extracted frames were cropped before being labeled, so that the pipe goes through the image, as Fig. 4 shows. The link for downloading MarinaPipe can be found in the REMARO GitHub.[2] We are releasing the original videos, the extracted frames, Tbl. I, and the frames' pixel-wise fine and coarse annotation for the pipeline class in the format of masks.

TABLE I: Details of MarinaPipe. (*Both* refers to fine and coarse labeling.)

Video	Selected frames	Frames with pipes	Annotation	Occlusions
1	236	43	Both	Yes
2	237	70	Both	No
3	260	2	Both	No
4	268	11	Both	Yes
5	266	45	Coarse	Yes
6	270	11	Coarse	Yes
7	186	17	Both	Yes

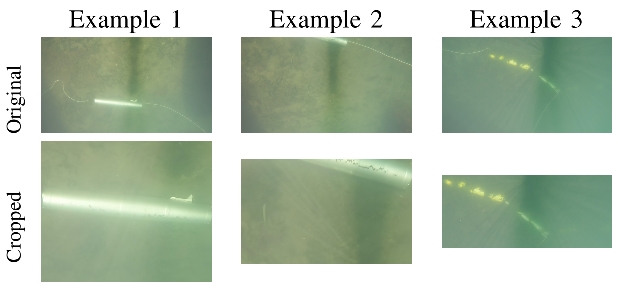


Fig. 4: Examples of how the video frames were cropped before being labeled. Top: Original frames. Bottom: Correspondent cropped frames.

[1]Based on the implementation from https://github.com/FrancescoSaverioZuppichini/SegFormer/

[2]https://github.com/remaro-network/MarinaPipe-dataset

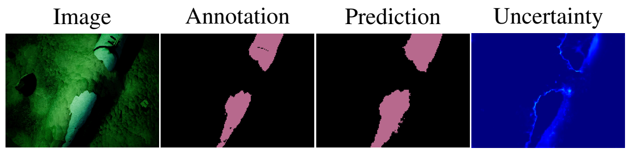


Fig. 5: Example of prediction on the test dataset of MIMIR using the model trained on MIMIR. The pipeline and background color are different from Fig. 3 because MIMIR has annotation for many classes and we are only using the pipeline class. Everything that is not pipeline is defined as background in this paper. For the uncertainty plot, calculated as the mutual information, the warmer colors represent higher values.

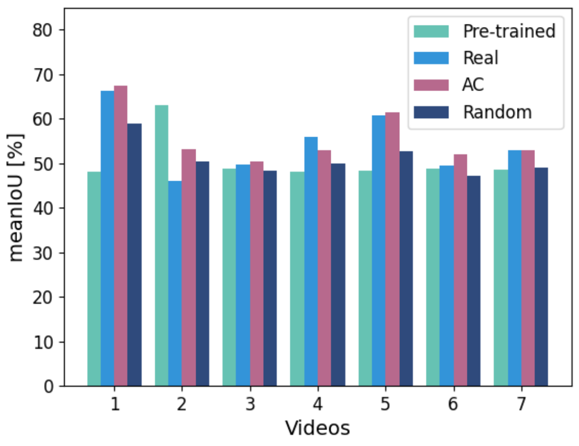


Fig. 6: Experimental results: meanIoU for the classes pipeline and background. Results were calculated using the coarse annotation as ground truth.

IV. Results

The sections below present the results obtained from our experiment for overcoming the sim-to-real gap. It also includes a study about the influence of data augmentation, freezing and unfreezing the decoder layers, the learning rate values, and the type of annotation used (fine or coarse).

A. Pre-training with MIMIR

From the SandPipe images selected for this study, 90% were reserved for training, 5% for validating, and the other 5% for testing SegFormer. From the data reserved for training, part of the images containing only background were eliminated to diminish the imbalance between this class and pipeline. After training, the model obtained 88.80% mean intersection over union (meanIoU) on the test dataset, with 81.05% intersection over union (IoU) for the pipeline class and 96.56% for the background. Figure 5 showcases an example of prediction using the pre-trained SegFormer on the MIMIR test subset.

B. Fine-tuning with real data

The experimental results are presented in Fig. 6, where the legends refer to the following set-ups:

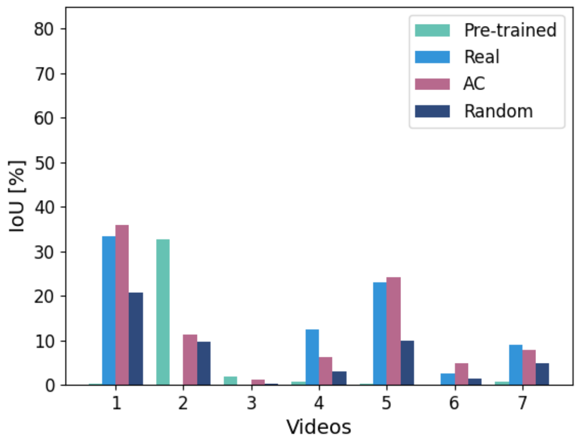


Fig. 7: Experimental results: IoU of pipeline. Results were calculated using the coarse annotation as ground truth.

- *Pre-trained* the model trained only on the MIMIR dataset at Sec. IV-A.
- *Real* the model was trained with the data from video 1, Tbl. I, and validated with video 7. Only part of the frames not containing pipelines were used, for diminishing the class imbalance. As augmentation, it was only applied resizing, cropping and flipping.
- *Random* the model was pre-trained with MIMIR, model from Sec. IV-A, and then fine-tuned with ca. 45% of images randomly selected from videos 1 and 7 for training and validation, respectively. The fine annotation was used as ground truth.
- *AC* the model training and fine-tuning was done as in *Random*, however the images were chosen with active learning, instead of at random.

The meanIoU results in Figure 6 were calculated using the coarse annotation as ground truth. For both *random* and *AC*, the same random augmentation techniques from Sec. III-A were applied to the training and validation images used to fine-tune the model.

For selecting new images with active learning, *AC*, we set $S = 3.0$ in Eq. (4). This parameter choice resulted in 110 images selected from MarinaPipe for training and 79 for validation. The same number of images were selected for training and validating the *random* model.

We found that fine-tuning the model with real images always reduces the sim-2-real gap. Figure 6 and Figure 7 show that active learning (AC) consistently outperforms random selection (Random). The results of the pipeline class in Figure 7 leave room for improvement. The pipeline's low IoU may be due to the dataset's complex patterns, which may require more annotated data to improve the model's performance. Even though the IoU for the pipeline is low, notice that the model fine-tuned with active learning (AC) can recognize this object, Fig. 8.

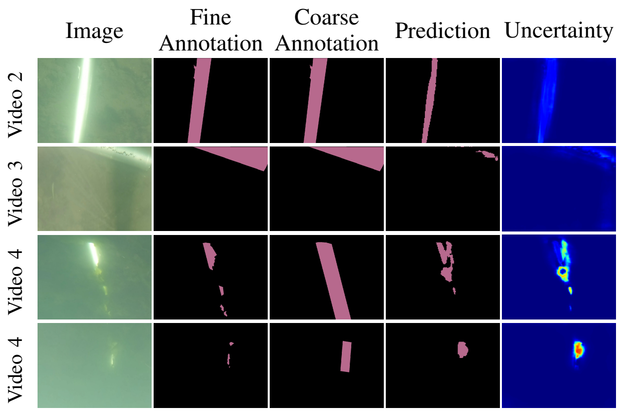


Fig. 8: Examples of predictions using the model fine-tuned with active learning, *AC*, and the respective mutual information uncertainties. Warmer colors represent higher uncertainty and reflect the models' difficulty when predicting pipelines.

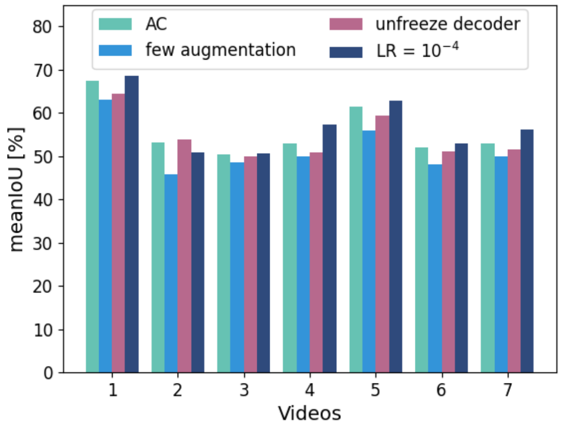


Fig. 9: These results analyze the influence of the choices for augmentation techniques, initial learning rate, and the option of freezing the model's decoder. *AC* refers to the model fine-tuned in Sec. IV-B.

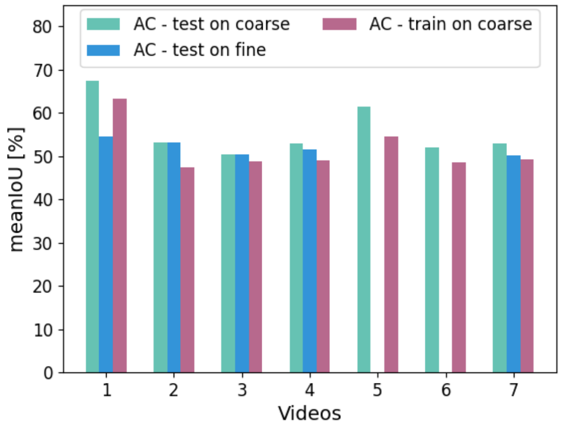


Fig. 10: *Test on coarse* and *test on fine* use the same model, trained in Sec. IV-B, but one is evaluated using the coarse annotation as in the referred section and the other using fine annotation. *Train on coarse* was trained and evaluated with coarse annotation. Notice that videos 5 and 6 do not have the fine annotation for performing the evaluation.

C. *Study of the augmentation, training and structure choices*

We fine-tune the model with very few data. To increase the model's performance, we used many augmentations and froze the entire encoder-decoder structure during the fine-tuning. Now, we present some comparisons between choices made during the fine-tuning phase. All the analyses were performed using the same frames used for training and validating the model *AC* in Sec. IV-B.

1) Data augmentation: Instead of using all the augmentations listed above, we now only apply resizing, cropping, and flipping to the data used for fine-tuning the model to test if so many augmentations were confusing the model.

2) Freeze vs. unfreeze the decoder: For analyzing the benefits of freezing the decoder during the fine-tuning phase, now only the encoder was frozen, allowing the decoder and the model's head to train.

3) Learning rate: For analyzing the initial learning rate choice during the fine-tuning, we test setting the initial value to 10^{-4} instead of 10^{-5}. Both the encoder and decoder were frozen, and only the head was allowed to train.

Figure 9 shows the results for the last three topics mentioned. As the figure shows, decreasing the amount of augmentation and unfreezing the decoder decreased the model's performance. Increasing the initial learning rate gave slightly better results. This was the only model fine-tuned with an initial learning rate equal to 10^{-4} in this study.

4) Test with fine annotation: As mentioned before, in Sec. IV-B, the models were fine-tuned using the fine annotation as ground truth; however, during the inference phase, the performance was analyzed using the coarse annotation. This choice has two reasons: (1) videos 5 and 6 only have the coarse annotation for evaluating the performance, and (2) apparently, the model learned how to extrapolate the fine annotation, and the results are better when compared to the coarse annotation.

5) Learning with coarse annotation: Since evaluating the performance in the coarse annotation gave better results, we wonder if training on coarse annotation would result in a better model. However, as Fig. 10 shows, the results of training using the coarse annotation were worse. We hypothesize that this is the case because the fine annotation gives a more "precise" label, and the model learns better to differentiate the pipeline from the rest of the image.

Figure 10 shows the results obtained for the tests in Sec. IV-C.4 and Sec. IV-C.5.

V. Conclusion

Active learning is more efficient in reducing the sim-to-real gap than fine-tuning with random images. The

MarinaPipe dataset has a lot of motion blur and uneven illumination, which could be the reason for the pipeline class's low IoU. Thus, MarinaPipe could be considered an open challenge to the underwater computer vision research community. SegFormer was trained with MIMIR from scratch and then fine-tuned with MarinaPipe. An interesting next test is to pre-train the model with a larger dataset, such as COCO, before using MIMIR, and evaluate if it would result in better IoU for the pipeline class. Even though it was demonstrated that SegFormer can be used with active learning, more tests should be performed to study the best positions to insert the dropout layers in this structure. In future work, we plan to select more images for labeling, and rerun the experiments with the additional annotated data.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 956200. We thank OceanScan - Marine Systems & Technology Lda for recording the real pipeline dataset. We thank Olaya Álvarez-Tuñón, László Antal, Martin Aubard, Sergio Quijano, Maria Costa, João fonseca and Renato Campos for the fruitful discussions during the development of this work.

References

[1] J. Feng, J. Lee, M. Durner, and R. Triebel, "Bayesian active learning for sim-to-real robotic perception," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10820–10827, 2022.

[2] R. Schettini and S. Corchs, "Underwater image processing: State of the art of restoration and image enhancement methods," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, p. 746052, 2010.

[3] A. Duarte, F. Codevilla, J. D. O. Gaya, and S. S. C. Botelho, "A dataset to evaluate underwater image restoration methods," in *OCEANS 2016 - Shanghai*, pp. 1–6, IEEE, 2016.

[4] J. Y. Chiang and Ying-Ching Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1756–1769, 2012.

[5] T. Li, S. Rong, L. Chen, H. Zhou, and B. He, "Underwater motion deblurring based on cascaded attention mechanism," *IEEE Journal of Oceanic Engineering*, 2022.

[6] W. Wang, R. Feng, J. Chen, Y. Lu, T. Chen, H. Yu, D. Z. Chen, and J. Wu, "Nodule-plus R-CNN and deep self-paced active learning for 3D instance segmentation of pulmonary nodules," *Ieee Access*, vol. 7, pp. 128796–128805, 2019.

[7] U. Gaur, M. Kourakis, E. Newman-Smith, W. Smith, and B. Manjunath, "Membrane segmentation via active learning with deep networks," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1943–1947, 2016.

[8] Y. Gal, *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

[9] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[10] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International conference on machine learning*, pp. 1183–1192, PMLR, 2017.

[11] I. C. Saidu and L. Csató, "Active learning with Bayesian UNet for efficient semantic image segmentation," *Journal of Imaging*, vol. 7, no. 2, p. 37, 2021.

[12] S. Xie, Z. Feng, Y. Chen, S. Sun, C. Ma, and M. Song, "Deal: Difficulty-aware active learning for semantic segmentation," in *Proceedings of the Asian conference on computer vision*, 2020.

[13] D. Sreenivasaiah, J. Otterbach, and T. Wollmann, "Meal: Manifold embedding-based active learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1029–1037, 2021.

[14] A. Rangnekar, C. Kanan, and M. Hoffman, "Semantic segmentation with active semi-supervised learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 5966–5977, 2023.

[15] O. Álvarez Tuñón, H. Kanner, L. Ribeiro Marnet, H. Xuy Pham, J. le Fevre Sejersen, Y. Brodskiy, and E. Kayacan, "MIMIR-UW: A multipurpose synthetic dataset for underwater navigation and inspection," *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023 (In publication).

[16] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, 2021.

[17] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, Jan. 2014.

[19] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[20] L. Ribeiro Marnet, Y. Brodskiy, S. Grasshof, and A. Wąsowski, "Uncertainty driven active learning for image segmentation in underwater inspection," in *Proceedings of the 4th International Conference on Robotics, Computer Vision and Intelligent Systems*, Springer Nature, Feb 2024.

Appendix E

Paper - Uncertainty Driven Active Learning for Image Segmentation in Underwater Inspection

Uncertainty Driven Active Learning for Image Segmentation in Underwater Inspection*

Luiza Ribeiro Marnet[1,2][0000−0001−6717−9306]
Yury Brodskiy[1][0009−0002−0445−8126]
Stella Grasshof[2][0000−0002−6791−7425]
Andrzej Wąsowski[2][0000−0003−0532−2685]

[1] EIVA a/s, Denmark
`{lrm, ybr}@eiva.com`
[2] IT University of Copenhagen, Denmark
`{stgr, wasowski}@itu.dk`

Abstract. Active learning aims to select the minimum amount of data to train a model that performs similarly to a model trained with the entire dataset. We study the potential of active learning for image segmentation in underwater infrastructure inspection tasks, where large amounts of data are typically collected. The pipeline inspection images are usually semantically repetitive but with great variations in quality. We use mutual information as the acquisition function, calculated using Monte Carlo dropout. HyperSeg is trained using active learning with an underwater pipeline inspection dataset of over 50,000 images. To allow reproducibility and assess the framework's effectiveness, the CamVid dataset was also utilized. For the pipeline dataset, HyperSeg with active learning achieved 67.5% meanIoU using 12.5% of the data, and 61.4% with the same amount of randomly selected images. This shows that using active learning for segmentation models in underwater inspection tasks can lower the cost significantly.

Keywords: Active learning · Computer vision · Underwater inspection.

1 Introduction

Computer vision plays a pivotal role in advancing automation across various applications, such as equipment inspection [37,2,17], autonomous driving [38,7,6], medical diagnoses [41,27,10], underwater debris detection [18,9], and underwater pipeline inspection [15,23]. However, the large amounts of annotated datasets required for training these models presents a major challenge. Annotating such datasets is time-consuming, expensive, or infeasible, especially in domains requiring expert knowledge.

* Partially supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 956200, REMARO.

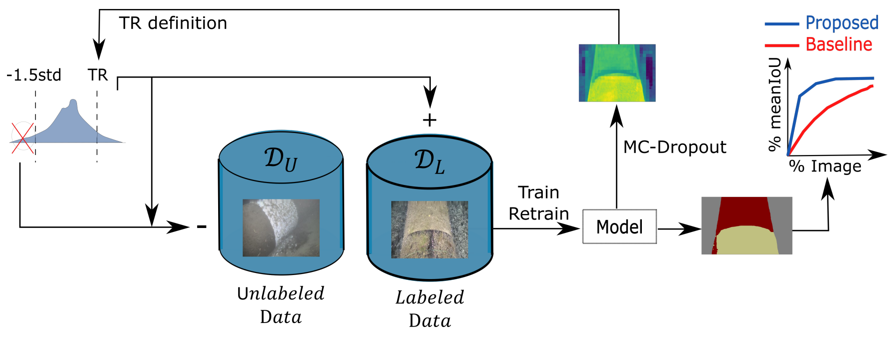


Fig. 1. Our active learning process: After pre-training the model with a small set of randomly chosen images, we start selecting images with uncertainty above a threshold TR (top-left) for training and validation of the model. In each iteration two new thresholds are defined: TR_t for training, and TR_v for validation images. We discard images with an uncertainty below the mean minus 1.5 standard deviations. In this figure, D_U and D_L are the pools of unlabeled and labeled images, respectively. The signs '+' and '$-$' indicate that the newly selected images are removed from D_U and added to D_L.

Even though access to large data is important, the quality of the data is critical. Active learning focuses on selecting the smallest sample set that can be used to train a model to achieve the same performance as when training with the entire dataset [30,5]. This typically involves training the model with few initial samples and selecting additional samples for labeling and retraining the model iteratively [5]. It is known that random selection usually does not perform well.

Epistemic uncertainty [1] can be used to identify samples that are most informative for training deep learning models. It measures the model's confidence in its predictions based on its level of familiarity with the input data [1,21]. By identifying samples that induce high epistemic uncertainty, the most informative samples can be labeled and added to the training dataset, potentially reducing the labeling effort while improving the model's performance.

In this paper, we investigate the use of epistemic uncertainty for image selection in the field of computer vision applications, cf. Fig. 1, more specifically to enhance the capabilities of autonomous robotic systems. We employ two distinct datasets, one for street view image segmentation, a crucial component in the domain of autonomous vehicle navigation, and the other for segmenting images captured during underwater robotics missions. Segmenting underwater images is a challenging task that can aid in, e.g., underwater autonomous vehicle path tracking, and is especially important for visual inspection of underwater equipment and structures. We demonstrate the performance of the active learning method with epistemic uncertainty when training HyperSeg [25] with real underwater RGB images from various missions and locations. To allow reproducibility, we also apply our approach to the street view dataset CamVid [4,3].

The results for the underwater images are specially important. These images come from a huge pool of images for pipeline inspection, and are semantically similar, but vary in quality due to factors like low and non-uniform illumination, color degradation, low contrast, blurring, and hazing [32,11,8]. Labeling these images requires specialized expertise and constitutes a significant cost for the growing but still small underwater automation industry. Selecting key images representing the different image qualities, reduces the required amount of training data, resulting in lower cost and human effort for labeling, which is valuable for innovating companies in this sector.

To the best of our knowledge, this is the first paper applying active learning for real underwater images. Our contributions include:

– A systematic study of the method against a random selection baseline, showing 6.2% gain in the meanIoU over the baseline in the underwater dataset.
– An active learning framework that uses a threshold for selecting new images instead of the usual fixed percentage of images at each active learning cycle.
– An implementation and reproduction package allowing to reproduce the results on the CamVid dataset using DenseNet and HyperSeg (the underwater imagery is unfortunately security-sensitive and cannot be released).

2 Related Work

In active learning with classical machine learning methods, it is common to select new samples using a metric for capturing uncertainty and another for measuring similarity. The former identifies samples for which the model is unsure about the predicted output, and is used for selecting the most informative samples for the model to learn from. The last aids in selecting representative samples and preventing the selection of redundant samples with similar information.

Thus, the first step in developing an active learning framework is to decide on an uncertainty metric. While the softmax values are often used as a measure that reflects the model's confidence for classification tasks, it has been demonstrated that misclassified samples can have high softmax values [26], for example, when an input out of the distribution of the training dataset is used during inference [13]. Therefore, other methods of capturing uncertainty should be used.

The uncertainty of the predictions, called predictive uncertainty, can be decomposed into epistemic and aleatoric [1]. Epistemic uncertainty reflects the model's lack of knowledge about the input's pattern, while aleatoric uncertainty reflects the data quality itself, e.g. noise caused by the sensor that captures the data. Since the goal of active learning is to select new images that bring knowledge that the model lacks for training, the important uncertainty in this scenario is the epistemic. Methods like Monte Carlo Dropout (MC-Dropout), deep ensembles, and error propagation can be used to access this uncertainty [12].

MC-Dropout is a widely used method for modeling epistemic uncertainty in deep neural networks. During inference dropout layers are used and the same input sample is passed forward multiple times through the model. Since dropout is applied, each pass may produce a different output, and these outputs are used to assess the epistemic uncertainty. If the model is well-trained and the input is similar to what the model has seen during training, the outputs will be similar or identical. On the other hand, if the model does not have enough knowledge about an input, the outputs of each forward pass will be more varied, indicating high uncertainty. Different metrics, such as variation ratio, mutual information, total variance, margin of confidence, and predictive entropy, can be used for that [24].

Ensembles of networks can also be used to predict epistemic uncertainty. In this method, an input sample is passed through the model and the results of each network in the ensemble are used to estimate the uncertainty [33]. Finally, the error propagation method [28] estimates the model uncertainty by propagating the variance of each layer to the output. This variance arises in layers such as dropout and batch normalization and is modified by the other layers of the model.

Deep active learning has been applied to tasks requiring costly labeling, such as medical image analysis. Softmax confidence with a single forward pass was used to evaluate the segmentation uncertainty for pulmonary nodules [36] and membrane [16] images, even if these may not be ideal for deep learning models. Other studies have proposed more reliable approaches. Using different subsets of training samples of biomedical images, a set of segmentation models was trained, and the uncertainty was measured as the variance between their outputs [40]. Different metrics, such as max-entropy and Bayesian active learning by disagreement (BALD), were calculated using MC-Dropout outputs for selecting new images to label for training skin cancer image classifier [14] and to segment medical images [31]. Moreover, a comparison between querying entire medical images and querying image paths concluded that the latter led to better models [22].

More recently, active learning was studied in the autonomous driving context. The Deeplabv3+ architecture with a Mobilenetv2 backbone was trained with uncertainty- and difficulty-driven image selection [39]. A further reduction of labeled pixels was obtained by querying image paths [34,29]. Even though these studies used entropy-based metrics for image selection, the metrics were calculated using a single pass softmax output and did not use MC-dropout.

In this work, we use MC-dropout to guide active learning in real-world problems. We first validate the method with DenseNet and HyperSeg for semantic segmentation [20] on the publicly available street view dataset CamVid [4,3]. We then use active learning with the better-performing model, HyperSeg [25], to obtain a new state-of-the-art model for real-time semantic segmentation in an underwater application. Our underwater dataset contains many images that are hard to segment even for human eyes. The dataset is unbalanced, with some classes appearing in only a few images and occupying a low percentage of pixels. The goal is to analyze the effectiveness of active learning in training the models with only a small percentage of the data and in learning to predict underrepresented classes.

3 Method

We now present an overview of our learning method: the models, the acquisition function, and the entire active learning framework.

Models. We employ two deep learning models for segmentation, DenseNet and HyperSeg. DenseNet [19], initially developed for classification, was later extended for segmentation [20]. We choose DenseNet due to its success in accessing epistemic uncertainty using MC-Dropout [21]. We utilize the lighter version, DenseNet-56.

HyperSeg is a state-of-the-art model based on the U-Net architecture [25]. In the decoder, it utilizes dynamic weights, that are generated based on both the input image and the spatial location. Because of its high mean intersection over union (meanIoU), high frames per second (FPS) rate, and the option to use dropout, we hypothesize that it is an appropriate choice. We use the version HyperSeg-S with efficientNet-B1 as the backbone [35], as it has achieved FPS of 38.0 and meanIoU of 78.4% in the original work. To the best of our knowledge HyperSeg has not been used in similar experiments to this date.

Acquisition Function. We calculate the epistemic uncertainty using MC-Dropout with T forward passes. The metric used was the mutual information, that for a segmentation problem with $\mathcal{C}$ classes is calculated for each pixel of the image as:

$$\mathcal{I} = \underbrace{\frac{-1}{\log_2(C)} \sum_{c=1}^{C} p_c^* \log_2(p_c^*)}_{\mathcal{H}} + \frac{1}{T\log_2(C)} \sum_{c=1}^{C} \sum_{t=1}^{T} p_{ct} \log_2(p_{ct}), \tag{1}$$

where $\mathcal{H}$ is the entropy, $p_t = (p_{1t}, ..., p_{Ct})$ is the softmax output for a single forward pass t, and $p^* = (p_1^*, ..., p_C^*)$ is the average prediction of the T passes. For each pixel, we predict one value p^*, and a class label c based on the maximum value of p_c^*. We divide the equation by $\log_2(C)$ to normalize the entropy between zero and 1.

Deep Active Learning Framework. In a real-world scenario the images are labeled after being queried. To reflect this, we only use the labels after the images are selected. Our active learning process begins by randomly selecting a constant $P\%$ of the images from both the training and validation sets in the first iteration. The model is then trained with these images. The epistemic uncertainty of each selected image is then calculated as the average uncertainty of each pixel:

$$EU_{\text{img}} = \frac{1}{N} \sum_{i=1}^{N} EU_i, \tag{2}$$

where EU_i is the epistemic uncertainty for pixel i, Eq. (1), of an image with N pixels. The next step is to calculate the thresholds TR_t and TR_v (Fig. 1) used to decide which images from the unlabeled training and validation datasets, respectively, should be selected for labeling. These thresholds are calculated as:

$$\mathrm{TR} = \overline{EU}_{\mathrm{img}} + S\sigma \tag{3}$$

where $\overline{EU}_{\mathrm{img}}$ and σ are the mean and standard deviation of EU_{img}, for the corresponding training and validation datasets, and S is a positive constant defined by the user. Bigger values of S result in querying fewer images. For the next iteration, the EU_{img} values of the images in the pools of unlabeled (remaining) training and validation data are computed. The images with EU_{img} above the respective thresholds, TR_t and TR_v, are selected for labeling. The model is then retrained with the new and previously selected images.

To accelerate the selection process, images with uncertainty values below 1.5 standard deviations below the mean are excluded from future selection. Since the model's predictions for these samples are relatively certain, using them to retrain the model is unlikely to improve performance. The value of 1.5 was empirically chosen and should be further studied in the future.

Training Details. DenseNet [20] was implemented from scratch and trained following the original paper, which includes pre-training and fine-tuning phases. The model was initialized with HeUniform in the first iteration, and with the weights of the best model from the previous iteration in subsequent iterations. RMSProp optimizer was used with a weight decay of $1e^{-4}$, and horizontal flip was applied to the training and validation datasets. Pre-training was performed using random crops of 224x224. The initial learning rate was $1e^{-3}$ in the first iteration and $1e^{-4}$ in the subsequent ones, with a learning rate decay of $1e^{-4}$. Fine-tuning was performed using 360x480 image resolution, an initial learning rate of $1e^{-4}$, and no learning rate decay. In each iteration, the pre-training process stopped after no improvement in validation meanIoU or validation loss for 150 consecutive epochs, and the fine-tuning process stopped after no improvement for 50 consecutive epochs.

For HyperSeg, we used the source code of the original authors.[3] The model's backbone was initialized with imagenet pre-trained weights in the first iteration and with the weights of the best model from the previous iteration in subsequent iterations. The initial learning rate was 0.001 in the first iteration and 0.00098 in subsequent iterations. The training of each iteration stopped after no improvement in validation meanIoU or validation loss for 10 consecutive epochs. The version chosen, HyperSeg-S, operates on 768x576 resolution.

Both models used a rate of 20% in each dropout layer utilized. DenseNet has a dropout layer after each convolutional layer. Hyperseg can add dropout layers at the end of the encoder and at the end of the hyper patch-wise convolution blocks in the decoder. For HyperSeg, we only used the dropout layer in the encoder. All experiments were developed using PyTorch.

[3] https://github.com/YuvalNirkin/hyperseg

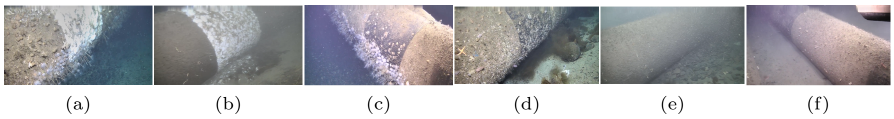


Fig. 2. Example of the pipeline dataset: (a-b) an anode and pipeline, (c) a pipeline and a field joint, (d) a field joint, boulders and pipeline, (e) a pipeline alone, and (f) a pipeline, field joint, and part of a vehicle in the top right (best viewed in color).

4 Experimental Evaluation

We discuss the optimal number of forward passes T used in MC-dropout, and the impact of applying active learning on the datasets *CamVid* [4,3], and *pipeline*.

Datasets. CamVid [4,3] is a street view dataset for segmentation. It contains video frames captured from the viewpoint of a driving car. The original dataset provides 369 images in the training subset, 100 in the validation subset, and 232 in the testing subset, in a resolution of 720×960. The half-resolution version consists of 367 images for training, 101 for validation, and 233 for testing. Labels for 32 hierarchical classes are provided from which we use 11: 'sky', 'building', 'column-pole', 'road,' 'sidewalk', 'tree', 'sign-symbol', 'fence', 'car', 'pedestrian', and 'bicyclist.'

The underwater dataset *pipeline* was provided by our industrial partner's customers and contains images from pipeline surveys. Due to security reasons,[4] the data is not publicly available. The entire dataset comprises 64,920 video frames recorded during surveys of infrastructure. Out of these frames, we reserved 8,896 for testing while the remaining 56,024 images have been randomly split into 70% for training and 30% for validation. The dataset has five classes: 'background', 'pipeline', 'field joint', 'anode', and 'boulder-and-survey vehicle', with respectively 99.9%, 77.1%, 20.3%, 21.0% and 15.5% of the training and validation images in each class. Figure 2 shows several examples. In the figure, some images have a more brownish background and others are more blueish. Image (e) suffers from higher turbidity than the other images. Marine flora grows on the pipelines in images (a), (c), and (d). Underwater images suffer from low and non-uniform illumination, color degradation, low contrast, blurring, and hazing [32,11,8]. The poor quality is due both to light attenuation and floating particles in marine environments, known as "marine snow" [32]. This causes haziness [11] and reduces visibility to below 20m, sometimes even below 5m. Also, the light is attenuated differently depending on the wavelength. The red light is attenuated faster, while blue is attenuated slower [8] resulting in the blueish hue of images.

We divide the two datasets into three subsets of images: training, validation, and testing. For the experiments with active learning, new images are iteratively selected from the pools of unlabeled training and validation images and added to the respective pools of labeled images, Fig. 3. For the test subset, the labels of

[4] E.g. https://en.wikipedia.org/wiki/2022_Nord_Stream_pipeline_sabotage

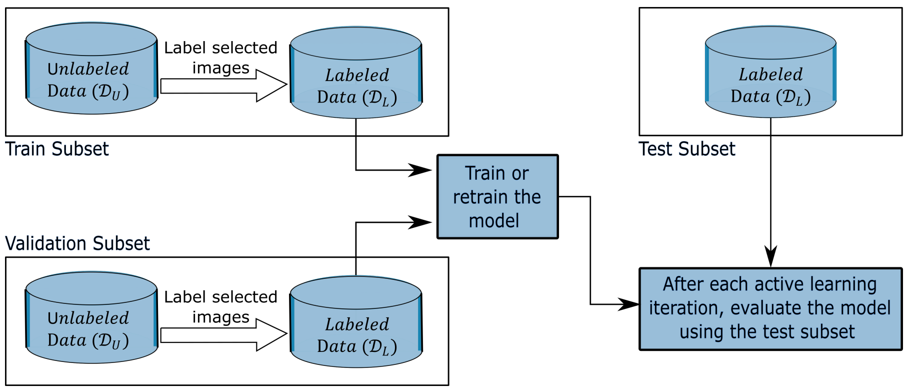

Unlabeled Data ($\mathcal{D}_U$)
Label selected images
Labeled Data ($\mathcal{D}_L$)
Train Subset
Labeled Data ($\mathcal{D}_L$)
Test Subset
Train or retrain the model
Validation Subset
Unlabeled Data ($\mathcal{D}_U$)
Label selected images
Labeled Data ($\mathcal{D}_L$)
After each active learning iteration, evaluate the model using the test subset

Fig. 3. The two datasets used, CamVid and pipeline, were divided into training, validation, and test subsets. For the active learning experiments, a few images from the training and validation subsets are initially labeled. At each iteration of the experiments, more images are selected, without replacement, from the respective pools of unlabeled images. The selected images are labeled and moved to the pools of labeled data. The entire test subset has labels since the beginning of the experiment.

all images have been used since the beginning, and this subset of images remains unchanged during all experiments and iterations. The flow diagram in this image shows how the images are moved and applied in each active learning iteration.

The Number of Forward Passes. In many papers the number of forward passes in MC-Dropout is set to $T = 50$, without clear justification. To address this shortcoming, we conducted a study to determine the optimal value of T. Since the epistemic uncertainty per image EU_{img}, Eq. (2), is used for querying new images for being labeled, this study should focus on how the number of passes T affects EU_{img}. We calculated $\overline{EU}_{\text{img}}$ of the dataset for $T = \{1, 2, 3, ..., 10\}$ and $T = \{20, 30, 40, ..., 200\}$. Each value of T was evaluated five times. The study was conducted using DenseNet-56 trained and validated with the complete subsets of CamVid. Figure 4 presents the results for the validation dataset. The graphs illustrate the mean and standard deviation of the results for the five repetitions. The graphs indicate that a stable mean value is achieved after T reaches 50, confirming the choice in published works. Furthermore, the standard deviation is over 100 times smaller than the mean, suggesting that 50 forward passes are enough to obtain a stable result with MC-Dropout. Henceforth in this paper, we define T as 50, in line with other works.

Active Learning for the CamVid Dataset. Figure 5 summarizes the design of experiment that we used. Starting with the experiment using DenseNet, at *step 1*, $P = 10\%$ images are selected for labeling, and the model is trained with the selected images at *step 2*. Two copies of the trained model, A and R, are saved. Model A will be further trained with active learning, and model R with randomly selected images. At *step 3*, the TR_t and TR_v are calculated with Eq. (3), in this

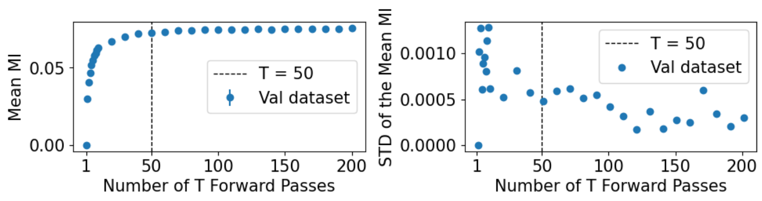


Fig. 4. Study of the number of forward passes T using the CamVid dataset and the model DenseNet. For each number of forward passes T, $\overline{EU}_{\text{img}}$ of the validation dataset of CamVid was calculated five times. The graph on the left shows the average of the five results obtained, and the graph on the right shows the standard deviation. Here $\overline{EU}_{\text{img}}$ was calculated using mutual information (MI).

experiment with $S = 1.0$, and new images are selected for labeling. After that, model A is retrained at *step 4*.

For evaluation purposes, a second model is trained using randomly selected images. At *step 5*, the same number of images N_{iter} from *step 3* is randomly selected, and model R is retrained at *step 6*. Notice that at each iteration, based on the calculated TR_t and TR_v, a different number of images is selected for training, N_{iter_train}, and validation, N_{iter_val}. Therefore, N_{iter} is different at each iteration. Notice also that the pools of labeled training and validation images, Fig. 3, generated with active learning at *step 3* of Fig. 5 are independent of the pools of images generated with the random selection at *step 5*, which means that the respective unlabeled pools are also independent.

At *Step 7* of Fig. 5, models A and R are evaluated using the test dataset, which remains the same in all iterations. After that, the two models are compared, analyzing the improvement of active learning against random selection. Finally, *step 8* evaluates the chosen stop criteria, e.g. if all images have been selected, no improvement in meanIoU on the test set is achieved, or few images have an epistemic uncertainty above the threshold. In this paper, for the CamVid experiments, we stop if very few images with uncertainty above the threshold are left. To observe if the performance of the models would suddenly improve, we keep the experiments running after unlabeled images stopped having uncertainty above the threshold.

Figure 6(a) is a result of *step 7* of Fig. 5, and presents the outcomes of the experiment on the CamVid dataset using DenseNet. The model's meanIoU stabilizes around 59% after iteration 12, using approximately 40% of the training data and slightly over 50% of the validation data, which represents around 41% of the data when the weighted mean is calculated for both subsets as shown in Fig. 6(a). After that, only one new image was selected for training and one for validation, indicating that additional images do not contribute significantly to the model's knowledge. Comparatively, training the model with the entire dataset yielded only a slightly higher meanIoU of 60.22% in our implementation. However, the model trained with uncertainty-selected images consistently outperformed the one trained with random images.

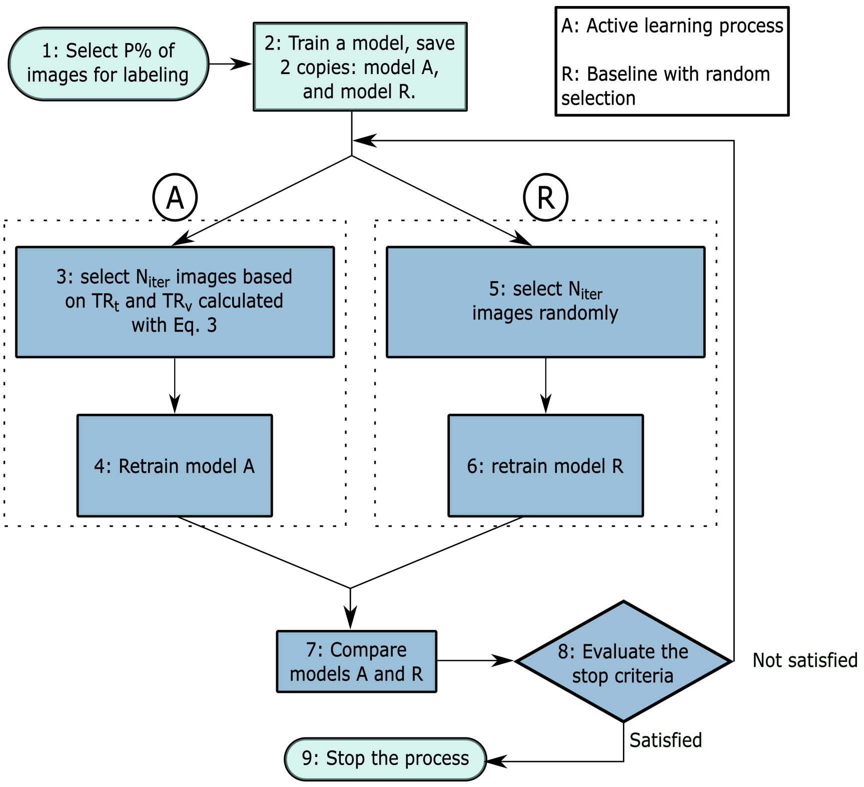

1: Select P% of images for labeling
2: Train a model, save 2 copies: model A, and model R.
A: Active learning process
R: Baseline with random selection
A
R
3: select N$_{iter}$ images based on TR$_t$ and TR$_v$ calculated with Eq. 3
5: select N$_{iter}$ images randomly
4: Retrain model A
6: retrain model R
7: Compare models A and R
8: Evaluate the stop criteria
Not satisfied
Satisfied
9: Stop the process

Fig. 5. The experiments begin with the random selection of $P\%$ of training and $P\%$ of validation images for training the model. At *step 3*, N$_{iter\_train}$ images are selected for training and N$_{iter\_val}$ for validation of model A. Based on the values of N$_{iter}$, defined at *step 3*, the same amount of images are randomly selected at *step 5*. The pools of labeled training and validation images for model A are composed of the $P\%$ initially selected images plus the images selected based on uncertainty in the iterations performed. The same applies to the pools used for training and validating model R, but in this case, the images are randomly selected. *Step 7* compares the performance of the two models. The whole process stops when the chosen criteria is achieved.

Additionally to DenseNet [21], we investigate the performance of HyperSeg, for which we start with $P = 10\%$ images and $S = 1.5$. The model trained with active learning outperformed the model trained with random images until around 25% of the images were queried, Fig. 6(b). After 25% of the images were selected, the performance of the models trained with images queried with uncertainty and randomly were very similar and stabilized around 69.0%. A possible reason for this behavior is that HyperSeg model has excellent generalization capabilities, requiring fewer data to achieve a stable meanIoU close to the performance obtained with the entire dataset. Finally, Fig. 7 and Tbl. 1 show results obtained

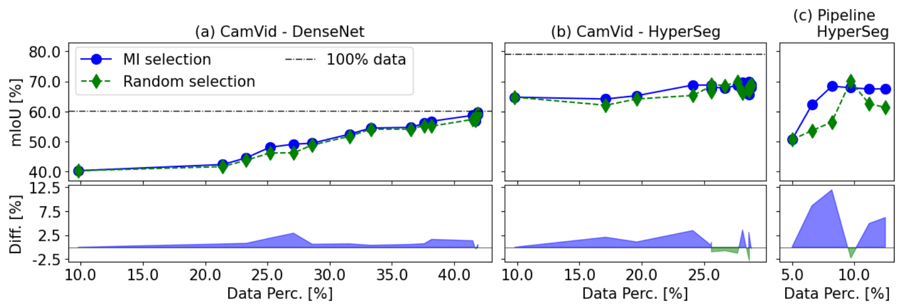


Fig. 6. Results of the active learning experiments. The bottom graphs show the difference in meanIoU between the model trained with active learning vs. trained with random images, where positive values means active learning prevails.

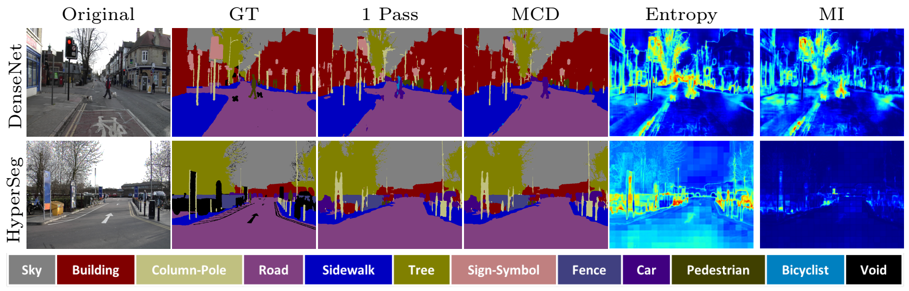


Fig. 7. Test segmentation results for the models trained with CamVid. GT is the ground truth. MCD is the average of the results obtained with MC-dropout. For the entropy and the mutual information (MI) plots, the warmer colors represent higher values. The entropy and MI heatmaps are normalized per experiment. (best viewed in color.)

with the models from the last iterations of the DenseNet and HyperSeg experiments. Table 1 also shows the performance of the models trained with the entire dataset.

In summary, applying active learning to CamVid confirmed the effectiveness of the active learning framework. It further reinforced that when data follows a consistent pattern, adding more samples does not necessarily improve the model's performance. As shown in Fig. 6(a-b), the meanIoU gain for DenseNet from the beginning to the end of the experiment is much more expressive than for HyperSeg. Even though more studies should be performed to analyze this behavior, we hypothesize that it regards the models' structure. HyperSeg apparently has a huge capability of generalization and does not require as much data as DenseNet to achieve the best performance possible.

Figure 8 compares our results with three recent deep active learning methods. Semi supervised semantic segmentation for active learning (S4AL) achieved around 61.4% meanIoU training on 13.8% of the data, which is 97% of the performance obtained with the entire dataset [29]. Similarly, Manifold Embedding-based Active Learning (MEAL) achieved 59.6% meanIoU, which is 81.6% of the

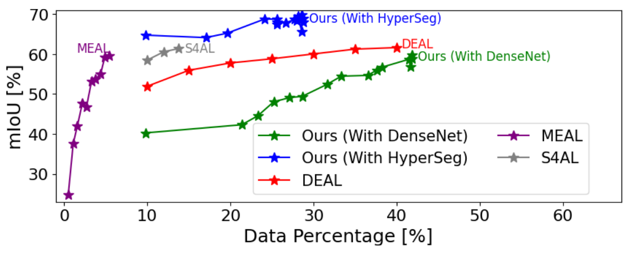


Fig. 8. Comparison of methods on CamVid. The values for DEAL, MEAL, and S4AL were manually extracted from the graphs in the original papers, hence are approximate. For each framework, the markers indicate when new images were labeled, and the models were (re-)trained in the respective original papers. (Best viewed online in color.)

overall performance, with just 5% of the images [34]. However, these frameworks queried patches of images instead of the whole image.

Difficulty-Aware Active Learning (DEAL) used image-level querying, and achieved 61.64% meanIoU, which is about 95% of the whole dataset performance, using 40.0% of the data [39]. Using DenseNet, we achieved 97.9% of the whole dataset's results with 41.9% of the data. With HyperSeg, we obtained 87.1% performance using only 28.9% of the data. As HyperSeg is a state-of-the-art model tailored for CamVid, the meanIoU is higher than for the other approaches. Notice also that the same framework requires a different percentage of data to obtain results close to the result obtained with 100% of the data when different model architectures are used, as we demonstrated using DenseNet and HyperSeg. Finally, when analyzing the results, it is important to remember that S4AL, DEAL, and MEAL are much more complex frameworks than ours. S4AL uses a teacher-student architecture for allowing semi-supervised training using pseudo labels. MEAL uses uniform manifold approximations and projection (UMAP) to learn a low dimensional embedding representation of the encoder output and uses K-Means++ to find the most representative between the most informative images sampled with entropy. DEAL adds to the CNN structure a probability attention module for learning semantic difficulty maps. Our framework consists of taking an out-of-the-box model without modifications and allowing dropout during the inference phase for using the MC-dropout and calculate the epistemic uncertainty.

Table 1. Results for the CamVid test subset. mIoU refers to meanIoU.

Experim.	% Data	Sky	Build.	Column	Road	Sidew.	Tree	Sign	Fence	Car	Pedes.	Bicyc.	mIoU
DenseNet	100	88.8	74.3	30.8	89.5	77.0	71.3	33.1	32.2	68.1	51.6	45.6	60.2
	41.9	91.9	77.6	28.2	92.0	77.6	74.1	26.5	26.0	70.0	47.5	37.3	59.0
HyperSeg	100	94.5	92.9	50.3	97.4	88.5	86.4	35.3	70.8	94.4	73.8	86.8	79.2
	28.8	85.8	84.5	38.5	94.3	7.2	79.9	47.7	55.5	88.7	46.5	61.0	69.0

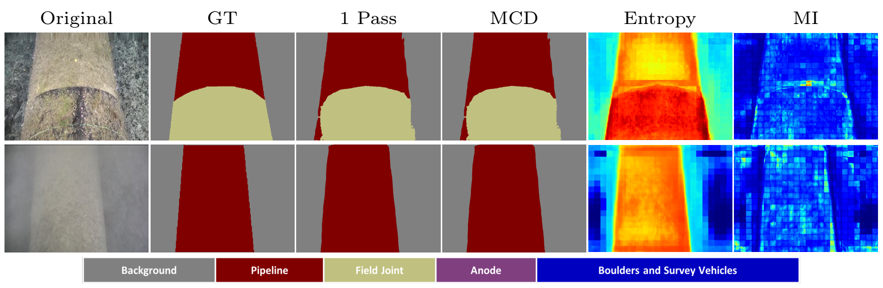


Fig. 9. Example results for pipeline test images. For the entropy and the mutual information plots, the warmer colors represent higher values (best viewed in color).

The Pipeline Dataset. The active learning process takes longer on this dataset, because it is larger than CamVid. We chose to run this experiment with HyperSeg, as it required less data and fewer iterations in the CamVid experiment, while achieving significantly higher meanIoU than DenseNet.

Figure 6(c) presents the results when starting with $P = 5\%$ of the images and $S = 1.5$. Because the pipeline dataset is huge, analyzing it entirely per iteration is time-consuming. First, the images are shuffled, then EU_{img} is calculated for each image. The image is selected or not based on the thresholds. The iteration stops when a pre-defined number of images is reached or the whole dataset is evaluated. For this experiment, we stop the active learning framework if the meanIoU does not improve significantly anymore. The model trained with uncertainty-based selection outperforms the baseline model trained with randomly selected images. The final meanIoU of the active learning model is 6.17% higher than the baseline. Except for iteration 4, the model trained with active learning was much more stable, presenting better and increasing performance across iterations. Table 2 presents the meanIoU for the test dataset for the final iteration models with uncertainty-based selection and with the baseline random selection. Note that for the most underrepresented class (boulder and survey vehicle) the model trained with uncertainty-based selection presents the most significant performance gain compared to the model trained with random images. Figure 9 showcases two examples of predictions for test images using the resulting model from the last iteration of this experiment. The entropy plots show higher values for the pipeline and the field joint, the relatively illegible classes, suggesting that using entropy as an acquisition function could yield good results.

Table 2. mIoU for the pipeline test dataset, for the models trained in the final iteration using 12.5% of the data. The percentages over headings indicate the amount of images containing each class in the entire training and validation datasets.

	99.9%	77.1%	20.3%	21.0%	15.5%	
Selection Criteria	Backg.	Pipe	F. Joint	Anode	B.&S. V.	mIoU
Uncertainty	97.0	84.6	66.3	31.1	58.6	67.5
Random (baseline)	96.2	80.0	61.3	29.0	40.3	61.4

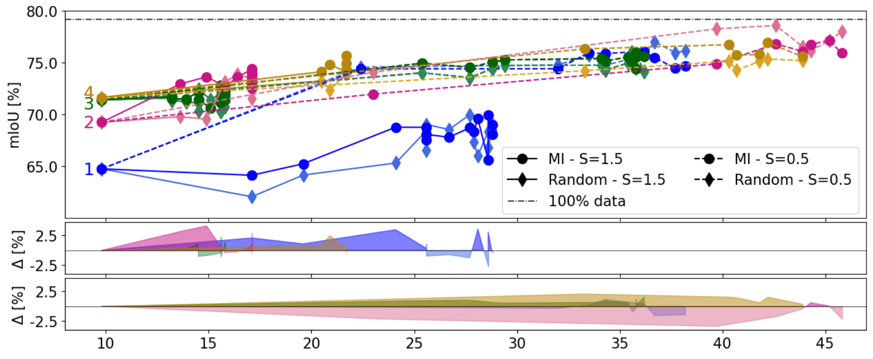


Fig. 10. Study of repeatability of results using the CamVid dataset and the model HyperSeg. Top: Four colors group run for different random initial sets of images. Below: mIoU differences **Δ** between the active learning and the baseline random selection models for $S = 1.5$ (middle) and $S = 0.5$ (bottom). Positive **Δ** means active learning prevails. (Best viewed online in color.)

In Fig. 6 the difference of meanIoU between the models trained with images selected based on uncertainty and the ones trained with randomly selected images is more significant in the pipeline experiment than in the CamVid experiments. The possible reason for that is that the pipeline dataset is much bigger and unbalanced than CamVid, making the selection of images more critical and challenging.

Repeatability. Learning depends on the initial set of images chosen for the first iteration. To test repeatability we reran the CamVid experiment with HyperSeg several times. We use CamVid because it is a smaller dataset than pipeline, allowing to run the experiments faster. While the original setup for CamVid with HyperSeg used $S = 1.5$ for thresholds in Eq. (3), we now used both $S = 1.5$ and $S = 0.5$, the latter selecting more images in each iteration. The top plot in Fig. 10 shows meanIoU results after each iteration for active learning and baseline random-selection models, grouping runs starting with different initial sets of images using four colors. As can be observed in the middle plot, active learning always prevails with $S = 1.5$. The bottom graph shows that for $S = 0.5$ the performance gain was smaller, and in the experiment number 2, in pink, random selection performed better. We hypothesize that as CamVid is a very small dataset and HyperSeg has a strong generalization capacity, when more data is selected the benefit of the uncertainty selection gets much lower. It remains an open question whether retraining the models from scratch at each iteration would prevent them from getting stuck in local minima yielding better performance. Notice that we reported in the previous paragraphs the performance for the experiment 1, in blue. The other experiments presented better meanIoU%. Experiment 4, in yellow, for example, achieved 75.7% meanIoU, 95.6% of the performance with the entire dataset, training on 21.7% of the data.

5 Conclusion

We demonstrated the effectiveness of active learning with epistemic uncertainty in an underwater infrastructure inspection task, using HyperSeg, a five-class dataset of more than fifty thousand images, and mutual information as the epistemic uncertainty measure. The HyperSeg structure did not need to be modified, making this method easy to implement. Using active learning for selecting the training images resulted in a model with 6.17% better meanIoU than a baseline model trained with the same number of random images. The model trained with active learning achieved 67.5% meanIoU using only 12.5% of the available data for training and validation. We observed that in the second iteration, the images queried attempted to compensate for the less represented classes in the dataset and the classes with lower performance in the previous iteration. This indicates that the approach helps with unbalanced datasets. Our experiment on the CamVid dataset, a small street view dataset with 11 semantic classes, suggested that the framework used is particularly effective for large datasets but may not provide a significant advantage for small datasets.

References

1. Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U.R., et al.: A review of uncertainty quantification in deep learning: Techniques, applications and challenges. Information Fusion **76**, 243–297 (2021)
2. Bouarfa, S., Doğru, A., Arizar, R., Aydoğan, R., Serafico, J.: Towards automated aircraft maintenance inspection. a use case of detecting aircraft dents using mask R-CNN. In: AIAA Scitech 2020 forum. p. 0389 (2020)
3. Brostow, G.J., Fauqueur, J., Cipolla, R.: Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters **30**(2), 88–97 (2009)
4. Brostow, G.J., Shotton, J., Fauqueur, J., Cipolla, R.: Segmentation and recognition using structure from motion point clouds. In: ECCV (1). pp. 44–57 (2008)
5. Budd, S., Robinson, E.C., Kainz, B.: A survey on active learning and human-in-the-loop deep learning for medical image analysis. Medical Image Analysis **71**, 102062 (2021)
6. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3d object detection for autonomous driving. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2147–2156 (2016). https://doi.org/10.1109/CVPR.2016.236
7. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1907–1915 (2017)
8. Chiang, J.Y., Ying-Ching Chen: Underwater image enhancement by wavelength compensation and dehazing. IEEE Transactions on Image Processing **21**(4), 1756–1769 (2012). https://doi.org/10.1109/TIP.2011.2179666, http://ieeexplore.ieee.org/document/6104148/
9. Chin, C.S., Bo Hui Neo, A., See, S.: Visual marine debris detection using yolov5s for autonomous underwater vehicle. In: 2022 IEEE/ACIS 22nd International Conference on Computer and Information Science (ICIS). pp. 20–24 (2022). https://doi.org/10.1109/ICIS54925.2022.9882484

10. Desai, M., Shah, M.: An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn). Clinical eHealth **4**, 1–11 (2021)
11. Duarte, A., Codevilla, F., Gaya, J.D.O., Botelho, S.S.C.: A dataset to evaluate underwater image restoration methods. In: OCEANS 2016 - Shanghai. pp. 1–6. IEEE (2016). https://doi.org/10.1109/OCEANSAP.2016.7485524, http://ieeexplore.ieee.org/document/7485524/
12. Feng, D., Harakeh, A., Waslander, S.L., Dietmayer, K.: A review and comparative study on probabilistic object detection in autonomous driving. IEEE Transactions on Intelligent Transportation Systems (2021)
13. Gal, Y.: Uncertainty in Deep Learning. Ph.D. thesis, University of Cambridge (2016)
14. Gal, Y., Islam, R., Ghahramani, Z.: Deep bayesian active learning with image data. In: International conference on machine learning. pp. 1183–1192. PMLR (2017)
15. Gašparović, B., Lerga, J., Mauša, G., Ivašić-Kos, M.: Deep learning approach for objects detection in underwater pipeline images. Applied Artificial Intelligence **36**(1), 2146853 (2022)
16. Gaur, U., Kourakis, M., Newman-Smith, E., Smith, W., Manjunath, B.: Membrane segmentation via active learning with deep networks. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 1943–1947 (2016). https://doi.org/10.1109/ICIP.2016.7532697
17. Guo, F., Qian, Y., Rizos, D., Suo, Z., Chen, X.: Automatic rail surface defects inspection based on mask R-CNN. Transportation research record **2675**(11), 655–668 (2021)
18. Hong, J., Fulton, M., Sattar, J.: Trashcan: A semantically-segmented dataset towards visual detection of marine debris. CoRR **abs/2007.08097** (2020), https://arxiv.org/abs/2007.08097
19. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
20. Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., Bengio, Y.: The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 11–19 (2017)
21. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf
22. Li, B., Alstrøm, T.S.: On uncertainty estimation in active learning for image segmentation. In: 2020 International Conference on Machine Learning: Workshop on Uncertainty and Robustness in Deep Learning (2020)
23. Medina, E., Petraglia, M.R., Gomes, J.G.R.C., Petraglia, A.: Comparison of cnn and mlp classifiers for algae detection in underwater pipelines. In: 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA). pp. 1–6 (2017). https://doi.org/10.1109/IPTA.2017.8310098
24. Milanés-Hermosilla, D., Trujillo Codorniú, R., López-Baracaldo, R., Sagaró-Zamora, R., Delisle-Rodriguez, D., Villarejo-Mayor, J.J., Núñez-Álvarez, J.R.: Monte carlo dropout for uncertainty estimation and motor imagery classification. Sensors **21**(21), 7241 (2021)

25. Nirkin, Y., Wolf, L., Hassner, T.: Hyperseg: Patch-wise hypernetwork for real-time semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4061–4070 (2021)
26. Oberdiek, P., Rottmann, M., Gottschalk, H.: Classification uncertainty of deep neural networks based on gradient information. In: IAPR Workshop on Artificial Neural Networks in Pattern Recognition. pp. 113–125. Springer (2018)
27. Polsinelli, M., Cinque, L., Placidi, G.: A light CNN for detecting COVID-19 from CT scans of the chest. Pattern recognition letters **140**, 95–100 (2020)
28. Postels, J., Ferroni, F., Coskun, H., Navab, N., Tombari, F.: Sampling-free epistemic uncertainty estimation using approximated variance propagation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2931–2940 (2019)
29. Rangnekar, A., Kanan, C., Hoffman, M.: Semantic segmentation with active semi-supervised learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5966–5977 (2023)
30. Ren, P., Xiao, Y., Chang, X., Huang, P.Y., Li, Z., Gupta, B.B., Chen, X., Wang, X.: A survey of deep active learning. ACM computing surveys (CSUR) **54**(9), 1–40 (2021)
31. Saidu, I.C., Csató, L.: Active learning with Bayesian UNet for efficient semantic image segmentation. Journal of Imaging **7**(2), 37 (2021)
32. Schettini, R., Corchs, S.: Underwater image processing: State of the art of restoration and image enhancement methods. EURASIP Journal on Advances in Signal Processing **2010**(1), 746052 (2010). https://doi.org/10.1155/2010/746052, https://asp-eurasipjournals.springeropen.com/articles/10.1155/2010/746052
33. Shamsi, A., Asgharnezhad, H., Jokandan, S.S., Khosravi, A., Kebria, P.M., Nahavandi, D., Nahavandi, S., Srinivasan, D.: An uncertainty-aware transfer learning-based framework for COVID-19 diagnosis. IEEE Transactions on Neural Networks and Learning Systems **32**(4), 1408–1417 (2021). https://doi.org/10.1109/TNNLS.2021.3054306
34. Sreenivasaiah, D., Otterbach, J., Wollmann, T.: Meal: Manifold embedding-based active learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1029–1037 (2021)
35. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
36. Wang, W., Feng, R., Chen, J., Lu, Y., Chen, T., Yu, H., Chen, D.Z., Wu, J.: Nodule-plus R-CNN and deep self-paced active learning for 3D instance segmentation of pulmonary nodules. Ieee Access **7**, 128796–128805 (2019)
37. Wang, Y., Liu, M., Zheng, P., Yang, H., Zou, J.: A smart surface inspection system using faster R-CNN in cloud-edge computing environment. Advanced Engineering Informatics **43**, 101037 (2020)
38. Xiao, X., Zhao, Y., Zhang, F., Luo, B., Yu, L., Chen, B., Yang, C.: Baseg: Boundary aware semantic segmentation for autonomous driving. Neural Networks **157**, 460–470 (2023)
39. Xie, S., Feng, Z., Chen, Y., Sun, S., Ma, C., Song, M.: Deal: Difficulty-aware active learning for semantic segmentation. In: Proceedings of the Asian conference on computer vision (2020)
40. Yang, L., Zhang, Y., Chen, J., Zhang, S., Chen, D.Z.: Suggestive annotation: A deep active learning framework for biomedical image segmentation. In: Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International

Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20. pp. 399–407. Springer (2017)

41. Zhou, X., Li, Y., Liang, W.: Cnn-rnn based intelligent recommendation for online medical pre-diagnosis support. IEEE/ACM Transactions on Computational Biology and Bioinformatics **18**(3), 912–921 (2020)

Bibliography

[1] OpenAI, “Chatgpt (december 2024 version),” https://openai.com, 2024, accessed: 2024-12-01.

[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[3] ——, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[4] Y. Wang, M. Liu, P. Zheng, H. Yang, and J. Zou, “A smart surface inspection system using faster R-CNN in cloud-edge computing environment,” *Advanced Engineering Informatics*, vol. 43, p. 101037, 2020.

[5] S. Bouarfa, A. Doğru, R. Arizar, R. Aydoğan, and J. Serafico, “Towards automated aircraft maintenance inspection. a use case of detecting aircraft dents using mask R-CNN,” in *AIAA Scitech 2020 forum*, 2020, p. 0389.

[6] F. Guo, Y. Qian, D. Rizos, Z. Suo, and X. Chen, “Automatic rail surface defects inspection based on mask R-CNN,” *Transportation research record*, vol. 2675, no. 11, pp. 655–668, 2021.

[7] X. Xiao, Y. Zhao, F. Zhang, B. Luo, L. Yu, B. Chen, and C. Yang, “Baseg: Boundary aware semantic segmentation for autonomous driving,” *Neural Networks*, vol. 157, pp. 460–470, 2023.

[8] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, “Multi-view 3d object detection network for autonomous driving,” in *Proceedings of the*

IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 1907–1915.

[9] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2147–2156.

[10] X. Zhou, Y. Li, and W. Liang, "Cnn-rnn based intelligent recommendation for online medical pre-diagnosis support," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 3, pp. 912–921, 2020.

[11] M. Polsinelli, L. Cinque, and G. Placidi, "A light CNN for detecting COVID-19 from CT scans of the chest," *Pattern recognition letters*, vol. 140, pp. 95–100, 2020.

[12] M. Desai and M. Shah, "An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn)," *Clinical eHealth*, vol. 4, pp. 1–11, 2021.

[13] J. Hong, M. Fulton, and J. Sattar, "Trashcan: A semantically-segmented dataset towards visual detection of marine debris," *CoRR*, vol. abs/2007.08097, 2020. [Online]. Available: https://arxiv.org/abs/2007.08097

[14] C. S. Chin, A. Bo Hui Neo, and S. See, "Visual marine debris detection using yolov5s for autonomous underwater vehicle," in *2022 IEEE/ACIS 22nd International Conference on Computer and Information Science (ICIS)*, 2022, pp. 20–24.

[15] B. Gašparović, J. Lerga, G. Mauša, and M. Ivašić-Kos, "Deep learning approach for objects detection in underwater pipeline images," *Applied Artificial Intelligence*, vol. 36, no. 1, p. 2146853, 2022.

[16] E. Medina, M. R. Petraglia, J. G. R. C. Gomes, and A. Petraglia, "Comparison of cnn and mlp classifiers for algae detection in underwater pipelines," in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2017, pp. 1–6.

[17] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *electronics*, vol. 8, no. 3, p. 292, 2019.

[18] S. E. Whang, Y. Roh, H. Song, and J.-G. Lee, "Data collection and quality challenges in deep learning: A data-centric ai perspective," *The VLDB Journal*, vol. 32, no. 4, pp. 791–813, 2023.

[19] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, "The effects of data quality on machine learning performance," *arXiv preprint arXiv:2207.14529*, 2022.

[20] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[21] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 305–311, 2020.

[22] M. Aubard, S. Quijano, O. Álvarez-Tuñón, L. Antal, M. Costa, and Y. Brodskiy, "Mission planning and safety assessment for pipeline inspection using autonomous underwater vehicles: A framework based on behavior trees," *arXiv preprint arXiv:2402.04045*, 2024.

[23] C. P. Langlotz, B. Allen, B. J. Erickson, J. Kalpathy-Cramer, K. Bigelow, T. S. Cook, A. E. Flanders, M. P. Lungren, D. S. Mendelson, J. D. Rudie *et al.*, "A roadmap for foundational research on artificial intelligence in medical imaging: from the 2018 nih/rsna/acr/the academy workshop," *Radiology*, vol. 291, no. 3, pp. 781–791, 2019.

[24] A. R. Munappy, J. Bosch, H. H. Olsson, A. Arpteg, and B. Brinne, "Data management for production quality deep learning models: Challenges and solutions," *Journal of Systems and Software*, vol. 191, p. 111359, 2022.

[25] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[26] P. Oberdiek, M. Rottmann, and H. Gottschalk, “Classification uncertainty of deep neural networks based on gradient information,” in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2018, pp. 113–125.

[27] R. Schettini and S. Corchs, “Underwater image processing: state of the art of restoration and image enhancement methods,” *EURASIP journal on advances in signal processing*, vol. 2010, pp. 1–14, 2010.

[28] A. Duarte, F. Codevilla, J. D. O. Gaya, and S. S. C. Botelho, “A dataset to evaluate underwater image restoration methods,” in *OCEANS 2016 - Shanghai*. IEEE, 2016, pp. 1–6.

[29] C. Chen, W. Xie, Y. Huang, X. Yu, and X. Ding, “Weakly-supervised man-made object recognition in underwater optimal image through deep domain adaptation,” in *International Conference on Neural Information Processing*. Springer, 2018, pp. 311–322.

[30] L. Jin and H. Liang, “Deep learning for underwater image recognition in small sample size situations,” in *OCEANS 2017 - Aberdeen*, 2017, pp. 1–4.

[31] X. Fan, P. Cao, P. Shi, X. Chen, X. Zhou, and Q. Gong, “An underwater dam crack image segmentation method based on multi-level adversarial transfer learning,” *Neurocomputing*, vol. 505, pp. 19–29, 2022.

[32] M. O’Byrne, V. Pakrashi, F. Schoefs, and B. Ghosh, “Semantic segmentation of underwater imagery using deep networks trained on synthetic imagery,” *Journal of Marine Science and Engineering*, vol. 6, no. 3, p. 93, 2018.

[33] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya *et al.*, “A review of uncertainty quantification in deep learning: Techniques, applications and challenges,” *Information Fusion*, vol. 76, pp. 243–297, 2021.

[34] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, “A review and comparative study on probabilistic object detection in autonomous driving,” *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[35] D. Milanés-Hermosilla, R. Trujillo Codorniú, R. López-Baracaldo, R. Sagaró-Zamora, D. Delisle-Rodriguez, J. J. Villarejo-Mayor, and J. R. Núñez-Álvarez, "Monte carlo dropout for uncertainty estimation and motor imagery classification," *Sensors*, vol. 21, no. 21, p. 7241, 2021.

[36] A. Shamsi, H. Asgharnezhad, S. S. Jokandan, A. Khosravi, P. M. Kebria, D. Nahavandi, S. Nahavandi, and D. Srinivasan, "An uncertainty-aware transfer learning-based framework for COVID-19 diagnosis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1408–1417, 2021.

[37] J. Postels, F. Ferroni, H. Coskun, N. Navab, and F. Tombari, "Sampling-free epistemic uncertainty estimation using approximated variance propagation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2931–2940.

[38] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf

[39] S. Budd, E. C. Robinson, and B. Kainz, "A survey on active learning and human-in-the-loop deep learning for medical image analysis," *Medical Image Analysis*, vol. 71, p. 102062, 2021.

[40] W. Wang, R. Feng, J. Chen, Y. Lu, T. Chen, H. Yu, D. Z. Chen, and J. Wu, "Nodule-plus R-CNN and deep self-paced active learning for 3D instance segmentation of pulmonary nodules," *Ieee Access*, vol. 7, pp. 128 796–128 805, 2019.

[41] U. Gaur, M. Kourakis, E. Newman-Smith, W. Smith, and B. Manjunath, "Membrane segmentation via active learning with deep networks," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 1943–1947.

[42] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Z. Chen, "Suggestive annotation: A deep active learning framework for biomedical image segmentation," in *Medical Image Computing and Computer*

Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20. Springer, 2017, pp. 399–407.

[43] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International conference on machine learning*. PMLR, 2017, pp. 1183–1192.

[44] I. C. Saidu and L. Csató, "Active learning with Bayesian UNet for efficient semantic image segmentation," *Journal of Imaging*, vol. 7, no. 2, p. 37, 2021.

[45] S. Xie, Z. Feng, Y. Chen, S. Sun, C. Ma, and M. Song, "Deal: Difficulty-aware active learning for semantic segmentation," in *Proceedings of the Asian conference on computer vision*, 2020.

[46] D. Sreenivasaiah, J. Otterbach, and T. Wollmann, "Meal: Manifold embedding-based active learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1029–1037.

[47] A. Rangnekar, C. Kanan, and M. Hoffman, "Semantic segmentation with active semi-supervised learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5966–5977.

[48] A. Parnami and M. Lee, "Learning from few examples: A summary of approaches to few-shot learning," *arXiv preprint arXiv:2203.04291*, 2022.

[49] N. Catalano and M. Matteucci, "Few shot semantic segmentation: a review of methodologies and open challenges," *arXiv preprint arXiv:2304.05832*, 2023.

[50] Z. Tian, X. Lai, L. Jiang, S. Liu, M. Shu, H. Zhao, and J. Jia, "Generalized few-shot semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 563–11 572.

[51] K. Huang, F. Wang, Y. Xi, and Y. Gao, "Prototypical kernel learning and open-set foreground perception for generalized few-shot semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 256–19 265.

[52] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41–65, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494618302813

[53] J. Y. Chiang and Y.-C. Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE transactions on image processing*, vol. 21, no. 4, pp. 1756–1769, 2011.

[54] C. Li, S. Anwar, and F. Porikli, "Underwater scene prior inspired deep underwater image and video enhancement," *Pattern Recognition*, vol. 98, p. 107038, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320319303401

[55] J. Perez, A. C. Attanasio, N. Nechyporenko, and P. J. Sanz, "A deep learning approach for underwater image enhancement," in *Biomedical Applications Based on Natural and Artificial Computing*, J. M. Ferrández Vicente, J. R. Álvarez-Sánchez, F. de la Paz López, J. Toledo Moreo, and H. Adeli, Eds. Cham: Springer International Publishing, 2017, pp. 183–192.

[56] G. Ramkumar, A. G, S. K. M, M. Ayyadurai, and S. C, "An effectual underwater image enhancement using deep learning algorithm," in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2021, pp. 1507–1511.

[57] X. Chen, P. Zhang, L. Quan, C. Yi, and C. Lu, "Underwater image enhancement based on deep learning and image formation model," 2021.

[58] C. Fabbri, M. J. Islam, and J. Sattar, "Enhancing underwater imagery using generative adversarial networks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7159–7165.

[59] C.-H. Yeh, C.-H. Huang, and C.-H. Lin, "Deep learning underwater image color correction and contrast enhancement based on hue preservation," in *2019 IEEE Underwater Technology (UT)*, 2019, pp. 1–6.

[60] T. Katayama, T. Song, T. Shimamoto, and X. Jiang, “Gan-based color correction for underwater object detection,” in *OCEANS 2019 MTS/IEEE SEATTLE*, 2019, pp. 1–4.

[61] M. J. Islam, Y. Xia, and J. Sattar, “Fast underwater image enhancement for improved visual perception,” *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 02 2020.

[62] E. H. Land, “The retinex theory of color vision,” *Scientific American*, vol. 237, no. 6, pp. 108–129, 1977. [Online]. Available: http://www.jstor.org/stable/24953876

[63] C. Wei, W. Wang, W. Yang, and J. Liu, “Deep retinex decomposition for low-light enhancement,” *CoRR*, vol. abs/1808.04560, 2018. [Online]. Available: http://arxiv.org/abs/1808.04560

[64] F. Han, J. Yao, H. Zhu, and C. Wang, “Underwater image processing and object detection based on deep cnn method,” *J. Sensors*, vol. 2020, 2020. [Online]. Available: https://doi.org/10.1155/2020/6707328

[65] J. Zhang, L. Zhu, L. Xu, and Q. Xie, “Research on the correlation between image enhancement and underwater object detection,” in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 5928–5933.

[66] C. Edge, M. J. Islam, C. Morse, and J. Sattar, “A generative approach for detection-driven underwater image enhancement,” *CoRR*, vol. abs/2012.05990, 2020. [Online]. Available: https://arxiv.org/abs/2012.05990

[67] L. Chen, Z. Jiang, L. Tong, Z. Liu, A. Zhao, Q. Zhang, J. Dong, and H. Zhou, “Perceptual underwater image enhancement with deep learning and physical priors,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, pp. 3078–3092, 2021.

[68] Z. Liu, B. Wang, Y. Li, J. He, and Y. Li, “Unitmodule: A lightweight joint image enhancement module for underwater object detection,” *Pattern Recognition*, vol. 151, p. 110435, 2024.

[69] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[70] Z. Cao, J. C. Principe, B. Ouyang, F. Dalgleish, and A. Vuorenkoski, “Marine animal classification using combined cnn and hand-designed image features,” in *OCEANS 2015-MTS/IEEE Washington*. IEEE, 2015, pp. 1–6.

[71] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” *Neural Information Processing Systems*, vol. 25, 01 2012.

[72] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[73] A. Mahmood, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, “Resfeats: Residual network based features for underwater image classification,” *Image and Vision Computing*, vol. 93, p. 103811, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885619301362

[74] P. Szymak, P. Piskur, and K. Naus, “remote sensing the effectiveness of using a pretrained deep learning neural networks for object classification in underwater video,” *Remote Sensing*, vol. 12, 09 2020.

[75] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[76] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[77] Y. Xu, Y. Zhang, H. Wang, and X. Liu, “Underwater image classification using deep convolutional neural networks and data augmentation,” in *2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2017, pp. 1–5.

[78] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[79] C.-C. Wang and H. Samani, “Object detection using transfer learning for underwater robot,” in *2020 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*. IEEE, 2020, pp. 1–4.

[80] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” 2018. [Online]. Available: https://arxiv.org/abs/1804.02767

[81] H. Yuan, S. Zhang, G. Chen, and Y. Yang, “Underwater Image Fish Recognition Technology Based on Transfer Learning and Image Enhancement,” *Journal of Coastal Research*, vol. 105, no. sp1, pp. 124 – 128, 2020. [Online]. Available: https://doi.org/10.2112/JCR-SI105-026.1

[82] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[83] L. Chen, F. Zhou, S. Wang, J. Dong, N. Li, H. Ma, X. Wang, and H. Zhou, “SWIPENET: object detection in noisy underwater images,” *CoRR*, vol. abs/2010.10006, 2020. [Online]. Available: https://arxiv.org/abs/2010.10006

[84] Z. Jiang and R. Wang, “Underwater object detection based on improved single shot multibox detector,” in *2020 3rd International Conference on Algorithms, Computing and Artificial Intelligence*, ser. ACAI 2020. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3446132.3446170

[85] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single shot MultiBox detector,” in *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007%2F978-3-319-46448-0_2

[86] A. F. Ayob, K. Khairuddin, Y. M. Mustafah, A. R. Salisa, and K. Kadir, “Analysis of pruned neural networks (mobilenetv2-yolo v2) for underwater object detection,” in *Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019*,

Z. Md Zain, H. Ahmad, D. Pebrianti, M. Mustafa, N. R. H. Abdullah, R. Samad, and M. Mat Noh, Eds. Singapore: Springer Singapore, 2021, pp. 87–98.

[87] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016.

[88] B. Fan, W. Chen, Y. Cong, and J. Tian, "Dual refinement underwater object detection network," in *ECCV*, 2020.

[89] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[90] W.-H. Lin, J.-X. Zhong, S. Liu, T. Li, and G. Li, "Roimix: proposal-fusion among multiple images for underwater object detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 2588–2592.

[91] W. Xu and S. Matzner, "Underwater fish detection using deep learning for water power applications," *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 313–318, 2018.

[92] H. Liu, P. Song, and R. Ding, "Towards domain generalization in underwater object detection," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 1971–1975.

[93] L. Chen, J. Dong, and H. Zhou, "Class balanced underwater object detection dataset generated by class-wise style augmentation," 2021.

[94] C. Liu, Z. Wang, S. Wang, T. Tang, Y. Tao, C. Yang, H. Li, X. Liu, and X. Fan, "A new dataset, poisson gan and aquanet for underwater object grabbing," 2021.

[95] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[96] T.-S. Pan, H.-C. Huang, J.-C. Lee, and C.-H. Chen, "Multi-scale resnet for real-time underwater object detection," *Signal, Image and Video Processing*, vol. 15, pp. 1–9, 07 2021.

[97] L. Chen, Z. Liu, L. Tong, Z. Jiang, S. Wang, J. Dong, and H. Zhou, "Underwater object detection using invert multi-class adaboost with deep learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[98] L. Zhang, X. Yang, Z. Liu, L. Qi, H. Zhou, and C. Chiu, "Single shot feature aggregation network for underwater object detection," *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 1906–1911, 2018.

[99] D. Zhang, L. Li, Z. Zhu, S. Jin, W. Gao, and C. Li, "Object detection algorithm based on deformable convolutional networks for underwater images," in *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)*, 2019, pp. 274–279.

[100] J. Zhang, L. Zhu, L. Xu, and Q. Xie, "Mffssd: An enhanced ssd for underwater object detection," in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 5938–5943.

[101] W. Chen and B. Fan, "Underwater object detection with mixed attention mechanism and multi-enhancement strategy," in *2020 Chinese Automation Congress (CAC)*, 2020, pp. 2821–2826.

[102] M. Gao, S. Li, K. Wang, Y. Bai, Y. Ding, B. Zhang, N. Guan, and P. Wang, "Real-time jellyfish classification and detection algorithm based on improved yolov4-tiny and improved underwater image enhancement algorithm," *Scientific Reports*, vol. 13, no. 1, p. 12989, 2023.

[103] W. Ji, J. Peng, B. Xu, and T. Zhang, "Real-time detection of underwater river crab based on multi-scale pyramid fusion image enhancement and mobilecenternet model," *Computers and Electronics in Agriculture*, vol. 204, p. 107522, 2023.

[104] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 713–13 722.

[105] X. Lv, A. Wang, Q. Liu, J. Sun, and S. Zhang, "Proposal-refined weakly supervised object detection in underwater images," in *Image and Graphics: 10th International Conference, ICIG 2019, Beijing, China, August 23–25, 2019, Proceedings, Part I 10*. Springer, 2019, pp. 418–428.

[106] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[107] L. Thampi, R. Thomas, S. Kamal, A. A. Balakrishnan, T. M. Haridas, and M. Supriya, "Analysis of u-net based image segmentation model on underwater images of different species of fishes," in *2021 International Symposium on Ocean Technology (SYMPOL)*. IEEE, 2021, pp. 1–5.

[108] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[109] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas, and S. S. C Botelho, "Underwater image segmentation in the wild using deep learning," *Journal of the Brazilian Computer Society*, vol. 27, no. 1, pp. 1–14, 2021.

[110] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[111] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1511.00561

[112] P. L. Drews, E. R. Nascimento, S. S. Botelho, and M. F. M. Campos, "Underwater depth estimation and image restoration based on single images," *IEEE computer graphics and applications*, vol. 36, no. 2, pp. 24–35, 2016.

[113] M. Xu, J. Su, and Y. Liu, "Aquasam: Underwater image foreground segmentation," *arXiv preprint arXiv:2308.04218*, 2023.

[114] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[115] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, “Semantic segmentation of underwater imagery: Dataset and benchmark,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1769–1776.

[116] J. Hong, M. Fulton, and J. Sattar, “Trashcan: A semantically-segmented dataset towards visual detection of marine debris,” *arXiv preprint arXiv:2007.08097*, 2020.

[117] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[118] X. Zhao, L. Jing, and Z. Du, “Research on image segmentation method of underwater pipeline oil leakage point,” in *2020 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, 2020, pp. 165–170.

[119] Z. He, L. Cao, J. Luo, X. Xu, J. Tang, J. Xu, G. Xu, and Z. Chen, “Uiss-net: Underwater image semantic segmentation network for improving boundary segmentation accuracy of underwater images,” *Aquaculture International*, pp. 1–14, 2024.

[120] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, “Ghostnet: More features from cheap operations,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1580–1589.

[121] J. Wang, F. Zha, and X. Bi, “Research on underwater image semantic segmentation method based on segnet,” in *Proceedings of the International Conference of Fluid Power and Mechatronic Control Engineering (ICFPMCE 2022)*, vol. 10. Springer Nature, 2023, p. 131.

[122] E. S. Saleh, T. M. Haridas, and M. Supriya, “Unsupervised image segmentation model based on w net architecture and conditional random field for underwater images,” in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1. IEEE, 2021, pp. 34–39.

[123] X. Xia and B. Kulis, “W-net: A deep model for fully unsupervised image segmentation,” 2017. [Online]. Available: https://arxiv.org/abs/1711.08506

[124] S. Raine, R. Marchant, B. Kusy, F. Maire, and T. Fischer, “Point label aware superpixels for multi-species segmentation of underwater imagery,” *arXiv e-prints*, pp. arXiv–2202, 2022.

[125] I. Alonso, M. Yuval, G. Eyal, T. Treibitz, and A. C. Murillo, “Coralseg: Learning coral segmentation from sparse annotations,” *Journal of Field Robotics*, vol. 36, no. 8, pp. 1456–1477, 2019.

[126] I. Alonso, A. Cambra, A. Munoz, T. Treibitz, and A. C. Murillo, “Coral-segmentation: Training dense labeling models with sparse ground truth,” in *Proceedings of the IEEE international conference on computer vision workshops*, 2017, pp. 2874–2882.

[127] M. C. R. LTER and P. Edmunds, “Mcr lter: Coral reef: Computer vision: Moorea labeled corals ver 3,” Environmental Data Initiative, 2019, accessed 2024-07-19. [Online]. Available: https://doi.org/10.6073/pasta/88dde0e68ab5232a470389f4bedd1892

[128] M. Bewley, A. Friedman, R. Ferrari Legorreta, N. Hill, R. Hovey, N. Barrett, O. Pizarro, W. Figueira, L. Meyer, R. Babcock, L. Bellchambers, M. Byrne, and S. Williams, “Australian sea-floor survey data, with images and expert annotations,” *Scientific Data*, vol. 2, 10 2015.

[129] M. S. Bewley, A. Friedman, R. Ferrari, N. Hill, R. Hovey, N. Barrett, O. Pizarro, W. Figueira, L. Meyer, R. Babcock, L. Bellchambers, M. Byrne, and S. Williams, “Benthoz-2015 public data set,” 2015. [Online]. Available: https://springernature.figshare.com/articles/dataset/BENTHOZ-2015_public_data_set/1524165/1

[130] G. Cutter, K. Stierhoff, and J. Zeng, “Automated detection of rockfish in unconstrained underwater videos using haar cascades and a new image dataset: Labeled fishes in the wild,” *Proceedings - 2015 IEEE Winter Conference on Applications of Computer Vision Workshops, WACVW 2015*, pp. 57–62, 02 2015.

[131] M. Jian, Q. Qi, J. Dong, Y. Yin, W. Zhang, and K.-M. Lam, “The ouc-vision large-scale underwater image database,” in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 2017, pp. 1297–1302.

[132] M. Pedersen, J. Bruslund Haurum, R. Gade, and T. B. Moeslund, "Detection of marine animals in a new underwater dataset with varying visibility," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 18–26.

[133] C. Liu, H. Li, S. Wang, M. Zhu, D. Wang, X. Fan, and Z. Wang, "A Dataset And Benchmark Of Underwater Object Detection For Robot Picking," *arXiv e-prints*, p. arXiv:2106.05681, Jun. 2021.

[134] K. Katija, B. Schlining, L. Lundsten, K. Barnard, G. Sainz, O. Boulais, B. Woodward, and K. C. Bell, "Fathomnet: An open, underwater image repository for automated detection and classification of midwater and benthic objects," *Marine Technology Society Journal*, vol. 55, no. 3, pp. 136–137, 2021.

[135] Z. Ma, H. Li, Z. Wang, D. Yu, T. Wang, Y. Gu, X. Fan, and Z. Luo, "An underwater image semantic segmentation method focusing on boundaries and a real underwater scene semantic segmentation dataset," 2021.

[136] L. Ribeiro Marnet, Y. Brodskiy, S. Grasshof, and A. Wąsowski, "Bridging the sim-to-real gap for underwater image segmentation," in *OCEANS 2024-Singapore*. IEEE, 2024.

[137] O. Álvarez Tuñón, L. R. Marnet, L. Antal, M. Aubard, M. Costa, and Y. Brodskiy, "Subpipe: A submarine pipeline inspection dataset for segmentation and visual-inertial localization," in *OCEANS 2024-Singapore*. IEEE, 2024.

[138] M. Waszak, A. Cardaillac, B. Elvesæter, F. Rødølen, and M. Ludvigsen, "Semantic segmentation in underwater ship inspections: Benchmark and data set," *IEEE Journal of Oceanic Engineering*, vol. 48, no. 2, pp. 462–473, 2022.

[139] A. Saleh, I. H. Laradji, D. A. Konovalov, M. Bradley, D. Vazquez, and M. Sheaves, "A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis," *Scientific Reports*, vol. 10, no. 1, p. 14671, 2020.

[140] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[141] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots, “One-shot learning for semantic segmentation,” 2017.

[142] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, “Segmentation and recognition using structure from motion point clouds,” in *ECCV (1)*, 2008, pp. 44–57.

[143] G. J. Brostow, J. Fauqueur, and R. Cipolla, “Semantic object classes in video: A high-definition ground truth database,” *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.

[144] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[145] A. Kendall and Y. Gal, “What uncertainties do we need in bayesian deep learning for computer vision?” *Advances in neural information processing systems*, vol. 30, 2017.

[146] A. Filos, S. Farquhar, A. N. Gomez, T. G. J. Rudner, Z. Kenton, L. Smith, M. Alizadeh, A. de Kroon, and Y. Gal, “A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks,” 2019. [Online]. Available: https://arxiv.org/abs/1912.10481

[147] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl, “Leveraging uncertainty information from deep neural networks for disease detection,” *Scientific reports*, vol. 7, no. 1, pp. 1–14, 2017.

[148] A. Jungo, R. Meier, E. Ermis, E. Herrmann, and M. Reyes, “Uncertainty-driven sanity check: Application to postoperative brain tumor cavity segmentation,” 2018. [Online]. Available: https://arxiv.org/abs/1806.03106

[149] A. Jungo, R. McKinley, R. Meier, U. Knecht, L. Vera, J. Pérez-Beteta, D. Molina-García, V. M. Pérez-García, R. Wiest, and M. Reyes, “Towards uncertainty-assisted brain tumor segmentation and survival prediction,” in *International MICCAI Brainlesion Workshop*. Springer, 2017, pp. 474–485.

[150] P. Tarling, M. Cantor, A. Clapés, and S. Escalera, "Deep learning with self-supervision and uncertainty regularization to count fish in underwater images," *PloS one*, vol. 17, no. 5, p. e0267759, 2022.

[151] B. Beckler, A. Pfau, M. Orescanin, S. Atchley, N. Villemez, J. E. Joseph, C. W. Miller, and T. Margolina, "Multilabel classification of heterogeneous underwater soundscapes with bayesian deep learning," *IEEE Journal of Oceanic Engineering*, 2022.

[152] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[153] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[154] O. Álvarez-Tuñón, H. Kanner, L. R. Marnet, H. X. Pham, J. le Fevre Sejersen, Y. Brodskiy, and E. Kayacan, "Mimir-uw: A multipurpose synthetic dataset for underwater navigation and inspection," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 6141–6148.

[155] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1411.4038

[156] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017. [Online]. Available: https://arxiv.org/abs/1706.05587

[157] K. Wada, "Labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme, 2016.

[158] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 12 077–12 090. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/64f1f27bf1b4ec22924fd0acb550c235-Paper.pdf

[159] F. S. Zuppichini, *Implementing SegFormer in PyTorch*, 5 2022. [Online]. Available: https://github.com/FrancescoSaverioZuppichini/SegFormer

[160] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[161] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 11–19.

[162] Y. Nirkin, L. Wolf, and T. Hassner, "Hyperseg: Patch-wise hypernetwork for real-time semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4061–4070.

[163] J. Feng, J. Lee, M. Durner, and R. Triebel, "Bayesian active learning for sim-to-real robotic perception," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 820–10 827.

[164] L. Cheng, Z. An, Y. Guo, M. Ren, Z. Yang, and S. McLoone, "Mmfsl: A novel multimodal few-shot learning framework for fault diagnosis of industrial bearings," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–13, 2023.

[165] X. Shi, S. Zhang, M. Cheng, L. He, X. Tang, and Z. Cui, "Few-shot semantic segmentation for industrial defect recognition," *Computers in Industry*, vol. 148, p. 103901, 2023.

[166] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330.

[167] Z. Lu, S. He, D. Li, Y.-Z. Song, and T. Xiang, "Prediction calibration for generalized few-shot semantic segmentation," *IEEE transactions on image processing*, vol. 32, pp. 3311–3323, 2023.

[168] S.-A. Liu, Y. Zhang, Z. Qiu, H. Xie, Y. Zhang, and T. Yao, "Learning orthogonal prototypes for generalized few-shot semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 319–11 328.

[169] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.