Novel Deep Learning Architectures for Marine and Aquaculture Applications

Thesis submitted by

Alzayat Saleh

on August, 2023, for the degree of

Doctor of Philosophy

at the

College of Science and Engineering
James Cook University

Supervisors

A/Prof. Mostafa Rahimi Azghadi

Associate Professor at the College of Science and Engineering at James Cook University, Australia.

Prof. Dean Jerry

Professor at the College of Science and Engineering at James Cook University, Australia.

Prof. Marcus Sheaves

Professor at the College of Science and Engineering at James Cook University, Australia.

Declaration and Statement of Access to This Thesis

Declaration

I, Alzayat Saleh, certify that:

The work presented in this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or institute of tertiary education.
This thesis contains a number of published research articles, all of which have been co-authored. The contribution of others is formally acknowledged in the *List of Publications Included in this Thesis* Section.

Statement of Access to This Thesis

I, the undersigned, the author of this work, understand that James Cook University will make this work available for use within the University, and via the Australian Digital Thesis Network, for use elsewhere.

I understand that as an unpublished work, a thesis has significant protection under the Copyright Act. I do not wish to place any restriction on access to this thesis. However, any use of its content must be acknowledged and could potentially be restricted by future patents.

Signed _____

Alzayat Saleh,
August, 2023

Acknowledgments

I am deeply grateful to my esteemed primary supervisor, Prof. Mostafa Rahimi Azghadi, for his exceptional guidance, unwavering support, and invaluable mentorship throughout my PhD journey. His expertise, dedication, and encouragement have been instrumental in shaping my research and professional development. I am truly fortunate to have him as my mentor and role model. I would also like to thank my secondary supervisors and mentors, Prof. Dean Jerry and Prof. Marcus Sheaves for their constructive suggestions, helpful advice and generous assistance in various aspects of my research. Their knowledge, experience and enthusiasm have enriched my learning and inspired me to pursue excellence.

I am profoundly grateful to my beloved children, Ahmed and Cady and I would like to dedicate this thesis to them: my son Ahmed, who has been my pillar of strength, my source of happiness and my best friend; and my daughter Cady, who has brought so much joy and light into my life. Words cannot express how much I love them and how grateful I am for their unconditional love and support. They are the reason why I embarked on this journey and why I persevered until the end. Likewise, I am deeply grateful to my best friend Adam Pete, who has supported, inspired and befriended me throughout my PhD journey. He has been a wonderful companion, a wise adviser and a loyal ally. He has shown me kindness, generosity and respect.

I appreciate the continuous encouragement, motivation, and belief in my abilities from my wife, my family and my friends. Your support has been a driving force behind my perseverance and determination to complete this PhD. I would also like to acknowledge the contributions of my colleagues, collaborators, and research partners who have shared their expertise, insights, and ideas, enriching my research and expanding my horizons. This includes the following great scientists: I. H. Laradji, D. Vazquez, P. Rodrguez, C. Lammie, C. Flavell, D. Nowrouzezahrai, D. Jones, M. Hasan, H. Raadsma, M. Khatkar, A. Kumar. And also, acknowledge the financial support from the Australian Government Research Training Program (RTP) Scholarship and Food Agility HDR Top-Up Scholarship, which enabled me to undertake this research.

Lastly, I would like to pay tribute to my late parents, who have always been my role models and inspirations. They have taught me the values of hard work, honesty and compassion. They have instilled in me the confidence and courage to pursue my dreams. They have loved me unconditionally and they are always in my heart I wish they could see my achievements. I deeply miss them and I hope they are proud of me.

Contribution of Others in this Thesis

This thesis contains a number of original research articles which have been published during my PhD candidature. These papers have been slightly modified to improve readability and cohesion in the form of a thesis document. Additionally, in all Chapters aside from the *Introduction* and *Conclusion and Future Work* Chapters, the terms **we** and **paper** are used to refer to all authors, collectively, and the corresponding chapter, respectively.

List of Publications Included in this Thesis

In this Section, a list of publications included in this thesis is presented, and all of the coauthors who contributed to the papers included in this thesis are formally acknowledged. A list of my publications during this PhD not included in this thesis is provided in the next Section.

1. Computer Vision and Deep Learning for Fish Classification in Underwater Habitats: A Survey

[1] Saleh, M. Sheaves, and M. Rahimi Azghadi, Computer vision and deep learning for fish classification in underwater habitats: A survey, Fish and Fisheries, vol. 23, no. 4, pp. 977999, 7 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/faf.12666
Location in thesis: Chapter 2
Student contribution: 90%. Developed the research idea and questions, conducted the literature review, generated figures, and wrote the paper.

2. Applications of Deep Learning in Fish Habitat Monitoring: A Tutorial and Survey

[2] Saleh, M. Sheaves, D. Jerry, and M. R. Azghadi, Applications of Deep Learning in Fish Habitat Monitoring: A Tutorial and Survey, 6 2022. [Online]. Available: http://arxiv.org/abs/2206.05394 Submitted to Expert Systems With Applications Journal
Location in thesis: Chapter 3
Student contribution: 90%. Developed the research idea and questions, conducted the literature review, generated figures, and wrote the paper.

3. Transformer-based Self-Supervised Fish Segmentation in Underwater Videos

[3] A. Saleh, M. Sheaves, D. Jerry, and M. R. Azghadi, Transformer-based Self- Supervised Fish Segmentation in Underwater Videos, IEEE, 6 2022. [Online]. Available: http://arxiv.org/abs/2206.05390 Submitted to Machine Vision and Applications Journal
Location in thesis: Chapter 4
Student contribution: 90%. Developed the research idea and questions, wrote and ran the experiments' code, generated figures, and wrote the paper.

4. How to Track and Segment Fish without Human Annotations: A Self-Supervised Deep Learning Approach

[4] A. Saleh, M. Sheaves, D. Jerry, and M. R. Azghadi, How to Track and Segment Fish without Human Annotations: A Self-Supervised Deep Learning Approach, 8 2022. [Online]. Available: https://arxiv.org/abs/2208.10662v1 Submitted to Pattern Analysis and Applications Journal
Location in thesis: Chapter 5
Student contribution: 90%. Developed the research idea and questions, wrote and ran the experiments' code, generated figures, and wrote the paper.

5. Adaptive Uncertainty Distribution in Deep Learning for Unsupervised Underwater Image Enhancement

[5] A. Saleh, M. Sheaves, D. Jerry, and M. R. Azghadi, Adaptive Uncertainty Distribution in Deep Learning for Unsupervised Underwater Image Enhancement, 12 2022. [Online]. Available: https://arxiv.org/abs/2212.08983v1 Submitted to Pattern Recognition Journal
Location in thesis: Chapter 6
Student contribution: 90%. Developed the research idea and questions, wrote and ran the experiments' code, generated figures, and wrote the paper.

6. MFLD-net: A Lightweight Deep Learning Network for Fish Morphometry using Landmark Detection

[6] Saleh, A., Jones, D., Jerry, D., and Azghadi, M. R. (2023). MFLD-net: a lightweight deep learning network for fish morphometry using landmark detection. Aquatic Ecology, 1-19. [Online]. Available: https://link.springer.com/article/10.1007/s10452-023-10044-8
Location in thesis: Chapter 7
Student contribution: 90%. Developed the research idea and questions, wrote and ran the experiments' code, generated figures, and wrote the paper.

7. Prawn Morphometrics and Weight Estimation from Images using Deep Learning for Landmark Localization

[7] A. Saleh, M. M. Hasan, H. W. Raadsma, M. S. Khatkar, D. Jerry, and M. R. Azghadi, Prawn Morphometrics and Weight Estimation from Images using Deep Learning for Landmark Localization, 2023. Submitted to Aquatic Ecology Journal
Location in thesis: Chapter 8
Student contribution: 90%. Developed the research idea and questions, wrote and ran the experiments' code, generated figures, and wrote the paper.

List of Publications Not Included in This Thesis

8. A Deep Learning Localization Method for Measuring Abdominal Muscle Dimensions in Ultrasound Images

[8] A. Saleh, I. H. Laradji, C. Lammie, D. Vazquez, C. A. Flavell and M. R. Azghadi, ”A Deep Learning Localization Method for Measuring Abdominal Muscle Dimensions in Ultrasound Images,” in IEEE Journal of Biomedical and Health Informatics, vol. 25, no. 10, pp. 3865-3873, Oct. 2021, doi: 10.1109/JBHI.2021.3085019.

9. Weakly Supervised Underwater Fish Segmentation using Affinity LCFCN

[9] Laradji, I.H., Saleh, A., Rodriguez, P. et al. Weakly supervised underwater fish segmentation using affinity LCFCN. Sci Rep 11, 17379 (2021). https://doi.org/10.1038/s41598-021-96610-2

Abstract

Marine and aquaculture applications pose unique challenges for computer vision due to the complex and dynamic underwater environment. Traditional methods based on handcrafted features and shallow models often fail to capture the rich information and variability in underwater videos and images. This thesis proposes novel deep learning architectures that leverage self-supervised learning, unsupervised learning, adaptive uncertainty distribution, and lightweight transformer models to address various tasks related to fish segmentation, trajectory tracking, image enhancement, landmark detection, weight estimation and morphometric analyses. Self-supervised learning is a technique that uses auxiliary tasks to learn useful representations from unlabelled data. Unsupervised learning is a technique that learns from data without any supervision or labels. Adaptive uncertainty distribution is a technique that models the uncertainty of the predictions and adapts it according to the input data. Lightweight transformer models are variants of the transformer architecture that reduce computational complexity and memory requirements while maintaining high performance. The proposed methods are evaluated on several public and private datasets of underwater videos and images collected from different sources and scenarios. The results demonstrate that the proposed methods achieve state-of-the-art performance in terms of accuracy, robustness and efficiency compared to existing methods. This thesis contributes to advancing the field of marine and aquaculture computer vision by providing novel solutions that can facilitate various applications such as fish monitoring, stock assessment, aquaculture management and environmental protection.

Contents

List of Tables

List of Figures

List of Acronyms

AI Artificial Intellegence

ANN Artifical Neural Network

API Application Programming Interface

CA Classification Accuracy

CNN Convolutional Neural Network

CPU Central Processing Unit

CV Computer Vision

DL Deep Learning

DNN Deep Neural Network

FC Fully Connected

FCN Fully Convolutional Network

GAME Grid Average Mean Absolute Error

GPU Graphics Processing Unit

IoMT Internet of Medical Things

IoT Internet of Things

JCU James Cook University

LCFCN Localisation-based Counting loss Fully Convolutional Network

MAE Mean Average Error

mAP Mean Average Precision

ML Machine Learning

MLP Multi-Layer Perceptron

NN Neural Network

PCA Principal Component Analysis

RAFT Recurrent All-Pairs Field Transforms

RF Random Forest

RNN Recurrent Neural Network

RUV Remote Underwater Video

SOTA State-Of-The-Art

SVM Support Vector Machine

TFLOPS Tera Floating Point Operations Per Second

Chapter 1

Introduction

1.1 Background and Motivation

Marine and aquaculture applications are vital for various purposes such as food production, biodiversity conservation, environmental monitoring, and economic development. However, these applications face many challenges [36] due to the complex and dynamic underwater environment, which poses difficulties in observing, measuring, and analyzing marine organisms and phenomena. Computer vision is a powerful tool that can help to overcome these challenges by providing automated and accurate methods for processing underwater videos and images. However, computer vision techniques developed for terrestrial or aerial scenarios are often not suitable or effective for underwater scenarios due to several challenges. Some of these challenges are:

- Low visibility, noise, distortion, illumination variation, occlusion, motion blur, and background clutter caused by light selective absorption and scattering as well as the use of artificial light.
- Quality degradation due to water turbidity, uneven illumination, monotonous colour, and complicated underwater background.
- Difficulty in estimating moving speed underwater due to water currents and buoyancy.
- Difficulty in obtaining accurate ground truth labels for training and evaluation due to limited human access and intervention.

These difficulties affect the experience of human perception and challenge the computer vision algorithms that are developed for terrestrial or aerial scenarios. Therefore, novel

theories and algorithms that can cope with these difficulties are needed for marine and aquaculture applications.

To address these challenges, this thesis proposes novel deep learning architectures that leverage self-supervised learning, unsupervised learning, adaptive uncertainty distribution, and lightweight transformer models to perform various tasks related to fish segmentation, trajectory tracking, image enhancement, landmark detection, weight estimation and morphometric analyses. These tasks are essential for various marine and aquaculture applications such as fish monitoring, stock assessment, aquaculture management and environmental protection.

Deep learning is a branch of machine learning that uses multiple layers of artificial neural networks to learn hierarchical representations from data. Deep learning has achieved remarkable success in various computer vision tasks such as object detection, face recognition, semantic segmentation and image generation. However, deep learning also faces some limitations such as the need for large amounts of labelled data, the lack of interpretability and generalization ability and the high computational complexity and memory requirements.

Self-supervised learning is a technique that uses auxiliary tasks to learn useful representations from unlabeled data. Self-supervised learning can reduce the need for manual annotation and exploit large amounts of unlabeled data available in marine and aquaculture domains. Unsupervised learning is a technique that learns from data without any supervision or labels. Unsupervised learning can discover hidden patterns and structures in data and generate realistic images or videos. Adaptive uncertainty distribution is a technique that models the uncertainty of the predictions and adapts it according to the input data quality and complexity. Adaptive uncertainty distribution can improve the performance and robustness of deep learning models by accounting for aleatoric and epistemic uncertainty sources. Aleatoric and epistemic are two types of uncertainty that can affect deep learning models. Aleatoric uncertainty arises from the inherent randomness or noise in the data, such as measurement errors, sensor noise, or natural variability. Epistemic uncertainty arises from the lack of data or knowledge about the true model, such as insufficient training samples, model misspecification, or parameter uncertainty. Both types of uncertainty can affect the performance and robustness of deep learning models and should be taken into account when making predictions or decisions based on them. Lightweight transformer models are variants of the transformer architecture that reduce computational complexity and memory requirements while maintaining high performance. Lightweight transformer models can capture long-range dependencies and global context better than

convolutional neural networks (CNNs), which rely on local features and fixed-size receptive fields.

The main motivation of this thesis is to design new deep learning methods that can handle the difficulties of processing underwater videos and images for marine and aquaculture applications. These methods use self-supervised learning, unsupervised learning, adaptive uncertainty distribution, and lightweight transformer models to learn from unlabeled data, model uncertainty, and capture global context. These methods aim to outperform existing methods in terms of accuracy, robustness and efficiency. These methods also aim to explain how and why deep learning works for underwater videos and images. The results of this research will contribute to the advancement of the field of marine and aquaculture computer vision by providing novel solutions that can improve the efficiency, accuracy, and sustainability of these industries. Moreover, the proposed methods may also have broader applications in other areas that require the analysis of visual data in challenging environments.

1.2 Research Problem

The research problem addressed in this thesis is the challenge of improving fish segmentation, trajectory tracking, underwater image enhancement, and fish morphometric analyses and weight estimation using self-supervised learning, unsupervised learning, and deep learning techniques. These challenges arise due to the complex nature of the underwater environment, which can include factors such as blurry images, cluttered backgrounds, and the resemblance between fish and their environment. Traditional approaches to these problems have relied on fully-supervised models that require manual annotations, limiting their generalizability. This thesis aims to overcome these limitations by proposing novel deep learning architectures that leverage self-supervised and unsupervised learning techniques to improve the accuracy and efficiency of fish segmentation, trajectory tracking, underwater image enhancement, fish morphometric analyses and weight estimation.

1.3 Research Questions

Based on the research problem, the following research questions are formulated:

1. How can self-supervised learning be used to improve fish segmentation and trajectory tracking in underwater videos?

2. How can unsupervised learning be used to enhance underwater images without any reference images or labels?

3. How can deep learning be used to improve fish morphometric analyses and weight estimation in an efficient and accurate manner?

These research questions are addressed in the subsequent chapters of this thesis. Each chapter presents a novel deep learning architecture that tackles one of the research questions and evaluates its performance on several public and private datasets of underwater videos and images collected from different sources and scenarios. The main contributions and findings of each chapter are summarized in the conclusion and future work chapter.

1.4 Original Contributions

The overall contribution of this thesis is the development of novel deep learning architectures that leverage self-supervised learning, unsupervised learning, adaptive uncertainty distribution, and lightweight transformer models to address various tasks related to fish segmentation, trajectory tracking, image enhancement, landmark detection, weight estimation and morphometric analyses in marine and aquaculture applications. These contributions are delivered through individual chapters that present specific methods and evaluate their performance on several datasets of underwater videos and images. The individual contributions of each chapter link together to form the overall contributions of the thesis, which advances the field of marine and aquaculture computer vision by providing novel solutions that can facilitate various applications such as fish monitoring, stock assessment, aquaculture management and environmental protection. Each chapter can be considered as a separate publication that presents a specific contribution to the overall research problem addressed in the thesis.

Specifically, this thesis comprises the following seven significant original research contributions to the field of marine and aquaculture computer vision:

1. In [1], a survey of Computer Vision and Deep Learning studies conducted between 2003-2021 on fish classification in underwater habitats was provided. Key concepts of Deep Learning were overviewed, while analyzing and synthesizing Deep Learning studies. The main challenges faced when developing Deep Learning for underwater image processing were discussed and approaches to address them were proposed. Insights were given into the marine habitat monitoring research domain

and future directions of Deep Learning for underwater image processing were suggested.

2. In [2], the application of DL techniques for underwater fish habitat monitoring was explored. A tutorial on the key concepts and steps of DL development was provided for marine scientists who want to learn and apply DL for their own problems. A comprehensive survey of existing DL methods for fish habitat monitoring tasks such as classification, counting, localisation, and segmentation was conducted. The performance and limitations of various DL techniques were compared using publicly available underwater fish datasets. Some open challenges and opportunities for future research in this domain were discussed. This paper aimed to bridge the gap between DL and underwater fish monitoring, and to facilitate the advancement of both fields.

3. In [3], a Transformer-based method that uses self-supervision for high-quality fish segmentation was introduced. The proposed model was trained on videos without any annotations to perform fish segmentation in underwater videos taken in situ in the wild. The model surpassed previous CNN-based and Transformer-based self-supervised methods and achieved performance relatively close to supervised methods on two new unseen underwater video datasets. The models compute-efficiency and great generalisability were also demonstrated.

4. In [4], a three-stage framework for robust fish tracking and segmentation was proposed. The framework used an optical flow model to generate pseudo labels using spatial and temporal consistency between frames. A self-supervised model refined the pseudo-labels incrementally. The refined labels were used to train a segmentation network. No human annotations were used during the training or inference. Experiments were performed on three public underwater video datasets to validate the method and demonstrate its effectiveness and robustness.

5. In [5], a novel framework called Uncertainty Distribution Network (UDnet) was proposed. UDnet learns to adapt to Uncertainty Distribution in its unsupervised reference map generation to produce enhanced output images. UDnet consists of three main parts: a statistically guided multi-colour space stretch module, a U-Net-like conditional variational autoencoder module, and a probabilistic adaptive instance normalization block. UDnet does not need manual human annotation and can learn with a limited amount of data to achieve state-of-the-art results. UDnet

was evaluated on eight publicly-available datasets and showed competitive performance compared to other state-of-the-art approaches.

6. In [6], a novel DL architecture called Mobile Fish Landmark Detection network (MFLD-net) was developed. MFLD-net can achieve keypoint detection accuracies on par or better than some of the state-of-the-art CNNs on a fish image dataset. MFLD-net uses convolution operations based on Vision Transformers. MFLD-net can achieve competitive or better results in low data regimes while being lightweight and suitable for embedded and mobile devices. MFLD-nets generalisation capabilities were also demonstrated.

7. In [7], a novel Deep Learning approach for automated weight estimation and morphometric analysis of prawns from images was proposed. The approach consisted of two main components: a feature extraction module that combined low-level and high-level features using the Kronecker product operation, and a landmark localization module that predicted the coordinates of key points on the prawn body using a localization network. Once these landmarks were extracted, weight was estimated using a weight regression module that estimated weight based on the extracted landmarks using a fully connected network. For morphometric analyses, we utilized the detected landmarks to derive five important prawn traits. Principal Component Analysis (PCA) was also used to identify landmark-derived distances, which were found to be highly correlated with shape features such as size, body length, and width. The approach was evaluated on a large-scale dataset of prawn images collected from various farms and environments. The experimental results demonstrated that the approach outperformed existing methods in terms of accuracy, robustness, and efficiency.

These contributions advance the state-of-the-art in marine and aquaculture computer vision by providing novel solutions that can facilitate various applications such as fish monitoring, stock assessment, aquaculture management and environmental protection.

1.5 Thesis Organization

As illustrated in Fig 1.1, this thesis is organized into nine chapters to convey all of the original research contributions in a coherent way. The current Chapter, i.e., the Introduction, highlighted in dark blue, introduces the research background and motivation.

In addition, research questions are formulated, and the key original contributions of this thesis are summarized.

The literature review component of this thesis is presented in Chapter 2 and Chapter 3, which are highlighted in orange in Fig 1.1. Chapter 2 of the thesis reviewed Deep Learning for fish identification in marine environments from 2003 to 2021. It discussed challenges and solutions for underwater image processing and offered insights into marine habitat monitoring and future directions.

Chapter 3 explored DL techniques for monitoring underwater fish habitats. It provided a tutorial for marine scientists and surveyed existing DL methods for fish habitat monitoring tasks. It compared the performance and limitations of various DL techniques and discussed open challenges and opportunities for future research. The chapter aimed to bridge the gap between DL and underwater fish monitoring, and to facilitate the advancement of both fields.

For the sake of clarity, each research question is highlighted using a different colour, and chapters are categorized using the aforementioned formulated research questions. In lieu of an abstract at the beginning of each Chapter, a short text passage is included introducing the Chapter and relating it to the formulated research questions.

As can be seen in Fig 1.1, Chapters 4, and 5 address the first research question. These chapters presented two novel methods for fish segmentation and tracking in underwater videos. The first used a Transformer-based model with self-supervision. The second used a three-stage framework with an optical flow model and self-supervised refinement. Both methods were validated on public datasets and demonstrated effectiveness and robustness.

Chapter 6 addresses the second research question. This chapter introduced a novel framework called UDnet that adapts to Uncertainty Distribution in its unsupervised reference map generation. UDnet consists of three main parts and does not require manual human annotation. It was evaluated on eight public datasets and showed competitive performance.

The third and last research question is addressed in Chapters 7 and 8. Both chapters proposed two novel DL architectures for fish landmark detection and prawn weight estimation. The first was MFLD-net, which used convolution operations based on Vision Transformers. The second was a novel DL approach for automated weight estimation and morphometric analysis of prawns. The approach was evaluated on a large-scale dataset and outperformed existing methods.

Finally, the thesis is concluded in Chapter 9, *Conclusion and Future Work*. In Fig 1.1, this Chapter is highlighted in the same colour as the Introduction to indicate a strong link/-

connection. In the Conclusion, the findings in other chapters are summarized with respect to the research questions formulated in the Introduction, and future research directions are discussed.

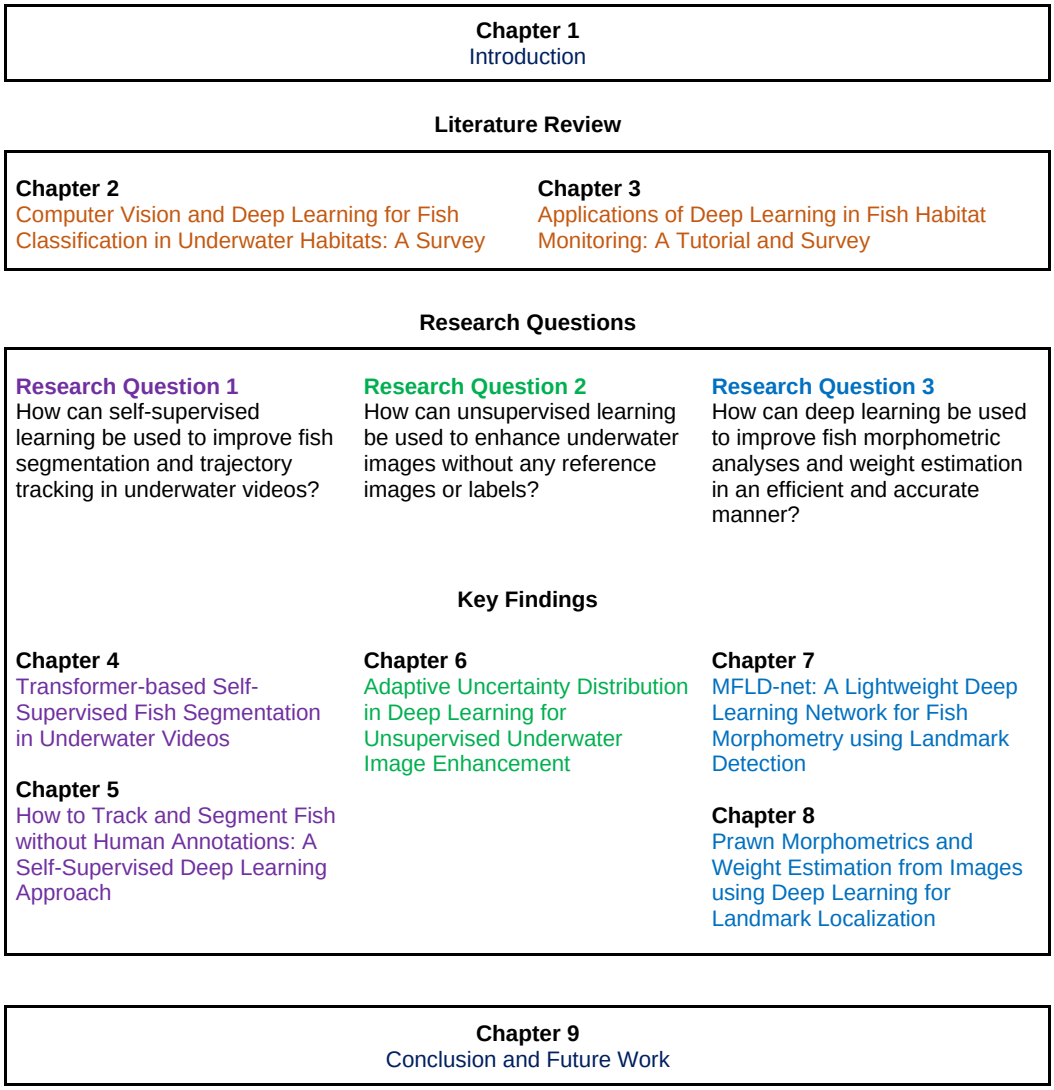

Figure 1.1: Thesis structure.

Chapter 2

Computer Vision and Deep Learning for Fish Classification in Underwater Habitats: A Survey

Marine scientists use remote underwater image and video recording to survey fish species in their natural habitats. This helps them get a step closer toward understanding and predicting how fish respond to climate change, habitat degradation, and fishing pressure. This information is essential for developing sustainable fisheries for human consumption, and for preserving the environment. However, the enormous volume of collected videos makes extracting useful information a daunting and time-consuming task for a human. A promising method to address this problem is the cutting-edge Deep Learning (DL) technology. DL can help marine scientists parse large volumes of video promptly and efficiently, unlocking niche information that cannot be obtained using conventional manual monitoring methods. In this Chapter, we first provide a survey of Computer Visions (CV) and DL studies conducted between 2003-2021 on fish classification in underwater habitats. We then give an overview of the key concepts of DL, while analyzing and synthesizing DL studies. We also discuss the main challenges faced when developing DL for underwater image processing and propose approaches to address them. Finally, we provide insights into the marine habitat monitoring research domain and shed light on what the future of DL for underwater image processing may hold. The results and analysis presented in this Chapter and the next Chapter led to the refined focus of this thesis on marine and aquaculture applications in Chapters 4-8.

2.1 Introduction

Understanding and modelling how fish respond to climate change, habitat degradation, and fishing pressure are critical for environmental protection, and are crucial steps toward ensuring sustainable natural fisheries, to support ever-growing human consumption [37]. Effective monitoring is a vital first step underpinning decision support mechanisms for identifying problems and planning actions to preserve and restore the habitats. However, there is still a gap between the complexity of marine ecosystems and the available monitoring mechanisms.

Marine scientists use underwater cameras to record, model, and understand fish habitats and fish behaviour. Remote Underwater Video (RUV) recording in marine applications [37] has shown great potential for fisheries, ecosystem management, and conservation programs [38]. With the introduction of consumer-grade high-definition cameras, it is now feasible to deploy a large number of RUVs or Autonomous Underwater Vehicles (AUVs) to collect substantial volumes of data and to perform more effective monitoring [39–41]. However, underwater habitats introduce diverse video monitoring challenges such as adverse water conditions, high similarity between fish species, cluttered backgrounds, and occlusions among fish. In addition, the volume of data generated by deployed RUVs and AUVs rapidly surpasses the capacity of human video viewers, making video analysis prohibitively expensive [42]. Moreover, humans are more prone to error than a well-designed machine-centred monitoring algorithm. Therefore, an automated, comprehensive monitoring system could significantly reduce labour expenses while improving throughput and accuracy, increasing the precision in estimates of fish stocks, fish distribution and biodiversity in general [43]. Implementing such systems necessitates effective Computer Vision (CV) processes. As a result, significant research has been conducted on implementing monitoring tools and techniques that build upon CV algorithms for determining how fish exploit various maritime environments and differentiating between fish species [44].

Figure 2.1: **Illustration of four typical types of CV tasks** From left: Image Classification (*i.e.* is there a fish in the image, or what type (class) of fish is in the image?), Object Detection/Localisation, Semantic Segmentation, Instance Segmentation.

In image analysis and CV domains, Deep Learning (DL) approaches have consistently produced state-of-the-art results in a variety of applications from agriculture [45] to medicine [46, 47] using Deep Neural Networks (DNNs) [48–50]. Notably, a video is inherently composed of images or frames, which are processed using image analysis techniques. Therefore, image- and video-based monitoring tasks can be done using DL models such as CNNs that receive an image (frame) as their input. Therefore, the methods mentioned for image-based tasks are useful for both images and videos.

Many of DNN-based approaches outperform conventional methods in marine applications, including ecological and habitat monitoring, using video trap data [51, 52]. DL is a technique that mimics how people acquire knowledge by continuous analysis of input data. The main drivers of DNN success over the past decade have been architectural progress by a large community of computer scientists, more powerful computers and processors, and access to massive amounts of data, which is critical for developing successful generalizable DL applications.

DNNs have been successfully employed in many CV applications such as object classification [53], identification [6], and segmentation as a result of the invention of CNN. CNN is a class of DNN, most commonly applied to visual analyses. For instance, CNNs have been successfully used for analysis of fish habitats [39, 42, 54]. In comparison to other image recognition algorithms, CNNs have the significant benefit that they require limited pre-processing. CNNs are not hand-engineered but uncover and learn hidden features in the data on their own. They learn level-by-level with various levels of abstraction. For instance, they learn simple shapes (edges, lines, etc.) in the first few layers, understand more sophisticated patterns in their next layers, and learn classes of objects in their final layers.

A putative challenge with CNNs is that they require a large number of images to be fully trained and generalise their learning to unseen scenarios. On the other hand, CNNs have an interesting and powerful feature that enables transfer of their learning and knowledge across different domains. This means that they can be fine-tuned to work on new datasets (e.g. fish datasets) other than the one that they have been trained on (e.g. general objects). However, fine-tuning with annotated datasets specific for a given domain implies cost/effort/time needed to generate the annotations, and also requires a larger set of data which may not always be available.

Equipping CV algorithms with the powerful learning and inference capabilities of CNNs can provide marine scientists and ecologists with powerful tools to help them better understand and manage marine environments. However, although DL, and its vari-

ants such as CNNs, have been applied to various applications across a multitude of domains [55–57], their use in conjunction with computer vision for marine science and fish habitat monitoring is not broadly appreciated, meaning they remain under utilised. To address this, in this paper, we introduce key concepts and typical architectures of DL, and provide a comprehensive survey of key CV techniques for underwater fish habitat monitoring. In addition, we provide insights into challenges and opportunities in the underwater fish habitat monitoring domain. It is worth noting that our article is written to provide a general and high-level, as opposed to detailed, introduction of deep learning and its relevant contexts for marine scientists. This is useful in understanding the follow-up discussions on the use of deep learning in the marine task of underwater fish classification.

Although a recent survey reviews deep learning techniques for marine ecology [58] and briefly discusses DL-based fish image analysis, to the best of our knowledge, no comprehensive survey and overview of deep learning with a specific focus on fish classification in underwater habitats currently exists. Our paper tries to address this gap and to facilitate the application of modern deep learning approaches into the challenging underwater fish images analysis and monitoring domains. We do this by comprehensively reviewing and analysing the literature providing information about the DL model the previous works have used, their training dataset, their annotation techniques, their performance and a comparison to other similar works. This detailed analysis is not provided in [58].

In addition, another survey [59] exists that focuses on five different tasks of classification, detection, counting, behaviour recognition, and biomass estimation. Compared to [59], we provide a different analysis and review of the literature because we mainly focus on the classification of fish in underwater images. Li and Du's work [59] fits mostly in the domain of aquaculture, while our paper is mostly a review of "fish classification techniques in underwater habitats" and the challenges they bring. Li and Du introduce a background to many different DL architectures, one of which is CNN, which is the focus of our paper. Also, the challenges and opportunities that Li and Du introduce are different to our paper, which is mainly about underwater fish classification in their natural habitat.

Furthermore, we provide a historical review of the CV and DL research using underwater cameras for fish classification, and analyse how their accuracy has evolved over years. This is not covered by previous works including [58, 59].

2.2 Background To Computer Vision and Machine Learning

Humans, have a natural ability to comprehend the three-dimensional structure of the world around us. Vision scientists [60] have spent decades attempting to understand how the human visual system functions [61]. Inspired by their findings, CV researchers [62–64] have also been working on ways to recover the 3D shape and appearance of objects from photos. The automatic retrieval, interpretation, and comprehension of useful information from a single image or collection of images can be referred to as CV. In another definition, CV is a field of Artificial Intellegence (AI) that focuses on training computers to detect, recognise, and understand images similarly to processes used by humans. This necessitates the development of logical and algorithmic foundations for automated visual understanding [65]. This understanding can include image classification, object localisation, object recognition, semantic segmentation, and instance segmentation, as shown in Figure 2.1. Today, computers with CV powers can extract, analyse, and interpret significant information from a single image or a sequence of images.

Despite this progress, the goal of making a computer to understand a picture at the same level as a two-year-old child remains unattainable. This is due, in part, to the fact that CV is an inverse problem in which we attempt to recover specific unknowns despite having inadequate knowledge to completely describe the solution. In CV applications, the cause is usually an exploration process, while the effects are the observed data. The corresponding forward problems then consist of predicting empirical data given complete knowledge of the exploration process. In some sense, solving inverse problems means "computing backwards", which is usually more difficult than forward problem solving [66].

The problem of backward computation was eased by the introduction of ML techniques more than 6 decades ago. However, in conventional ML approaches, the majority of complex features of the learning subject must be identified by a domain expert in order to decrease the complexity of the data and make patterns more evident for successful learning (see Figure 2.2-top). However, DL offered a fundamentally new method to ML. Most DL algorithms possess the ground-breaking ability of automatically learning high-level features from data with minimal or no human intervention (see Figure 2.2-bottom).

DL is based on neural networks, which are general-purpose functions that can learn almost any data type that can be represented by many instances. When you feed a neural network a large number of labelled instances of a certain type of data, it will be able to

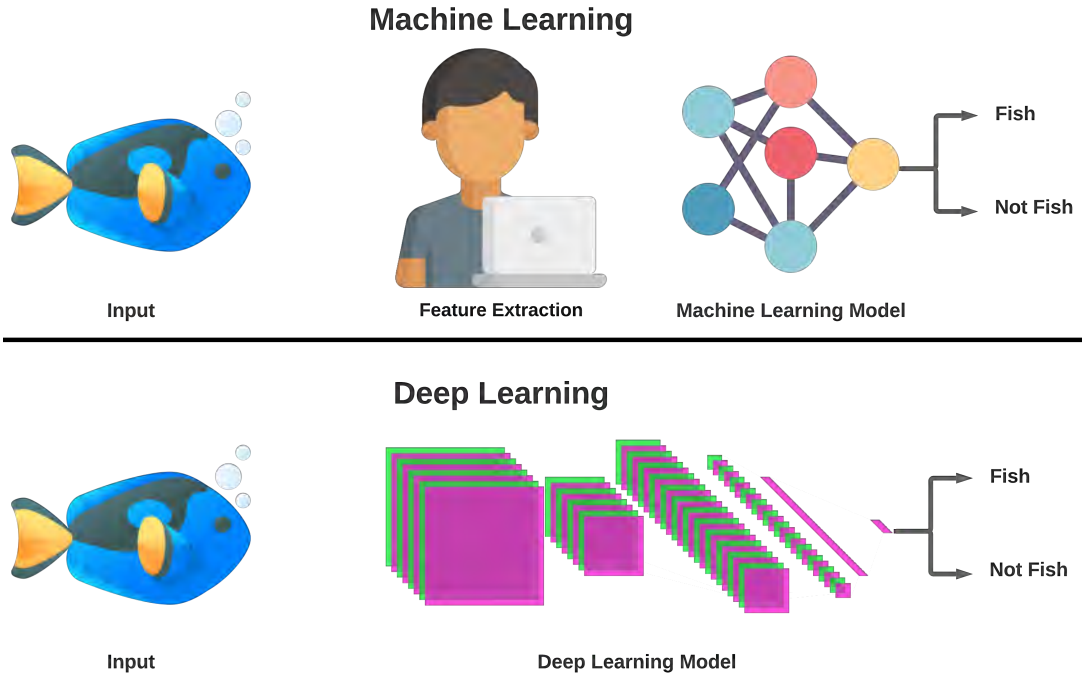

Figure 2.2: **Comparison between ML and DL.** In ML techniques, the features need to be extracted by domain expert while DL relies on layers of artificial neural networks to extract these features.

uncover common patterns between those examples and turn them into a mathematical equation that will assist in categorising future data. Empowered by this fundamental feature, DL and DNN have progressed from theory to practice as a result of advancements in hardware and cloud computing resources [47]. In recent years, DL approaches have outperformed previous state-of-the-art ML techniques in a variety of areas, with CV being one of the most notable examples.

Before the introduction of DL, the capabilities of CV were severely limited, necessitating a great deal of manual coding and effort. However, owing to improved research in DL and neural networks, CV is now able to outperform humans in several tasks related to object recognition and classification [67–70]. CV equipped with DL, is being used today in a wide variety of real-world applications, that include, but are not limited to:

- *Optical character recognition (OCR)* [71]: automatic number plate recognition and reading handwritten postal codes on letters;
- *Machine inspection* [72]: fast quality assurance inspection of components using stereo vision with advanced lighting to assess tolerance levels on aircraft wings or car body parts, or to spot flaws in steel castings using X-ray technology;

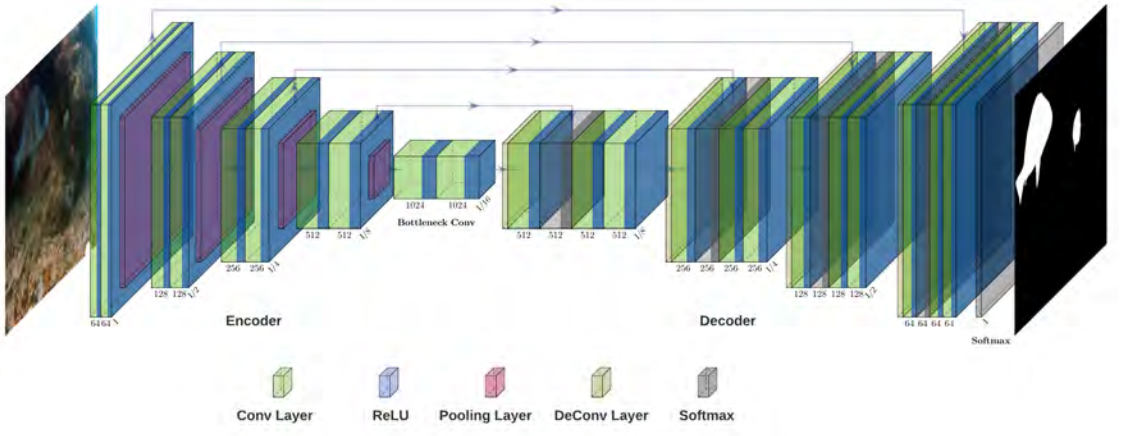

Figure 2.3: A popular CNN architecture, named UNET [32] is demonstrated. The first component of UNET is the encoder, which is used to extract features from the input image. The second component is the decoder that outputs per-pixel scores. The network is composed of five different layers including convolutional (Conv Layer), Rectified Linear Unit (ReLU), Pooling, Deconvolutional (DeConv), and Softmax.Here, the task of the DNN layers has been to give a high score to only the pixels in the input image that belong to the fish body, resulting in the demonstrated white blobs output, showing where the fish are.

- *Retail* [73]: object detection for automatic checkout lanes;
- *Medical imaging* [74]: registration of preoperative and intra-operative imaging or long-term analyses of human brain anatomy as they age;
- *Automotive safety* [75]: detection of unforeseen objects such as pedestrians on the street (e.g. fully autonomously driving vehicles);
- *Surveillance* [76]: Monitoring of trespassers, studies of highway traffic, and monitoring pools for drowning victims;
- *Fingerprint recognition and bio-metrics* [77]: For both automatic entry authentication and forensic software.

This demonstrates the significant impact of DL on CV and demonstrates its potential for marine visual analysis applications.

2.3 The evolution of Computer vision approaches to fish classification

The last two decades have witnessed the emergence of novel computer vision approaches for fish classification including the design and evaluation of complex algorithms that could not be applied before and became possible with the availability of sufficiently large data and the use of powerful Graphical Processing Units (GPUs). Here, we perform a systematic literature review of the evolution of computer vision applications and their different approaches over the past two decades.

2.3.1 Search and Selection Criteria

We systematically reviewed the literature for underwater fish classification using computer vision from 2003 to 2021. The search terms used included ”underwater fish classification”, ”Deep Learning”, ”Computer Vision”, ”Machine vision”. The databases searched included Wiley Online Library, IEEE Xplore, Elsevier/ScienceDirect, and ACM Digital Library. We believe that combining these four databases accurately represents global research on this topic.

We divided the search into two stages. First, we queried the databases for articles with the above-mentioned keywords in their titles and contents. Secondly, we independently reviewed the titles and abstracts of each article in order to check its relevance to our research topic. After the individual title and abstract reviews, we considered 64 articles for full-text reading. In the full-reading phase, we extracted information relevant to our research topic. In this phase, it became clear that 21 papers were not relevant to our work and therefore were excluded. This left us with 43 papers for fish classification, 26 of which were classical Computer Vision methods, and 17 Deep Learning papers. Figure 2.4 presents an overview of the methods used in the identified studies and classifies them into several groups, based on their classification algorithms that can be categorized into two general category of conventional CV, and modern DL models.

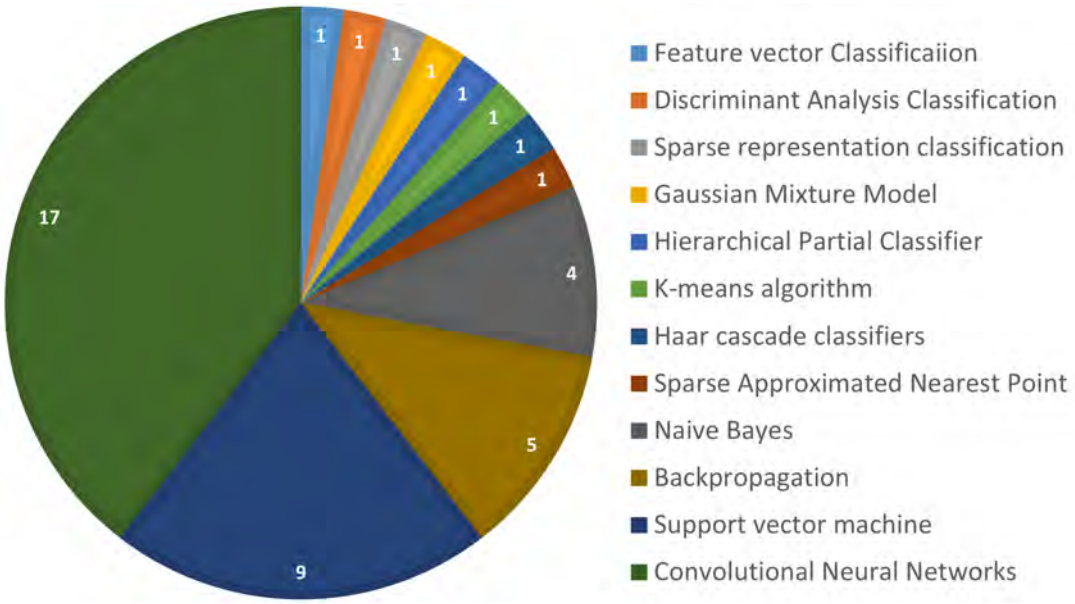

Figure 2.4: An overview of the methods used for fish classification using different Computer Vision techniques from 2003 to 2021. It is evident from the graph that DL and its CNNs have attracted more attention than classical ML methods.

2.3.2 The Evolution of Fish Classification Algorithms over Two Decades

The publication trend for fish classification studies is summarized in Fig. 2.5. The figure shows the cumulative number of publications and how the studies evolved over the past two decades. It is evident that the number of publications has been gradually increasing, but in 2016, when the first few studies using deep learning were combined with CV methods, the study numbers have seen the highest increase and a fast upward trajectory for a few years (2015-2019) after DL burgeoned in fish classification, and before slowing down.

Fig. 2.5 also shows the highest classification accuracy achieved in each year, as **a quality assessment metric**. It is evident that since 2016, when DL techniques were first proposed for fish classification, the accuracy has seen its highest value. At the same time, it can be seen that there are large differences in the accuracies achieved over years. The main reasons for this difference include (i) using different classification and CV methods, and (ii) using different fish image sources that were captured differently and in different environments. These bring huge variations among studies, such as different image resolutions and inconsistent resolutions and image qualities across time. For example, some fish image datasets are in grayscale [78–80], while others are in colour [81–83]. Some

datasets contain only images [80, 84], while others include videos [85–87]. Also, some datasets [88] used low-quality images from the internet, which negatively affects the accuracy, due to their wide range of resolutions, colours, and angles. They are also taken at random locations. Due to these factors in various studies, direct comparison of accuracy values is unfeasible, though the accuracy trend can be still observed in Fig. 2.5.

Figure 2.5: An overview of the publication trend and performance of an extensive range of fish classification Computer Vision (CV) and Deep Learning (DL) models from 2003 to 2021. Here the bars show the cumulative number of publications over years and the growth thereof, while the line graphs demonstrate the highest classification accuracy in each year in literature on the right-hand-side vertical axis.

Computer vision for fish classification in the early 2000s and up to 2016, when first DL works started, has been mainly to manually extract fish features and then build classifiers that recognize these features. These conventional studies are listed, in a chronological order, in Table 2.1. Although there are many existing models, most of the classical non-DL models are based on local and engineered features. These include works using Haar features [89], Scale-Invariant Feature Transform (SIFT) [90], and Histogram of Oriented Gradient (HOG) [91], which need hand-engineered algorithms. Because these algorithms are not suitable for recognizing images of untrained animals and cannot capture fish features from complex backgrounds, they usually use a large number of manually extracted samples to build classifiers.

As shown in Table 2.1, support vector machines [79, 84, 87, 88, 92–96] were one of the most commonly used classifiers for fish recognition, but they are prone to overfitting when trained with too many samples. This problem limits the scale of application. Another popular classification technique in early works is using a simple feed-forward shallow

neural network trained using backpropagation [97–101]. Although this technique can handle simple samples, it is difficult to scale because of the neural network shallow layers, which will be explained in the next Section. Naive Bayes [80–82, 102] have also been used to classify fish since the early 2000s and up to 2017. The technique does not require much training data, and as shown in Table 2.1 can reach good accuracy levels. Table 2.1 also shows some other CV classification techniques, which while not as popular as the above-mentioned methods, could demonstrate good performance. However, it should be noted that, most of the CV techniques in Table 2.1, were carefully engineered for their target datasets and are not capable of showing a similar performance level if used for another similar dataset. They will perhaps require an overhaul in their design, starting from manual feature engineering, to designing the detailed classification models.

In contrast, deep learning can extract features and perform classification tasks automatically. The features are invariant to data scaling, translation, rotation, and distortion. Because these features are better for classification, the classification performance can be better than that conventional CV tasks using manually designed features. Also, DL classification models, compared to traditional CV one, usually require a simpler redesign procedure to work on a new similar dataset, due to the ability to extract features on their own.

Although DL emerged in 2012 [103], its first use for underwater fish classification was in 2016 [68]. After that, 16 other works also used DL and its CNNs, as shown in Fig. 2.4, to develop models that learn features from large amounts of data without manual interference. These studies have shown that, by using deep learning, some of the usual fish image classification challenges such as image noise reduction, classification of difficult or rare-seen fish, and classifying small fish, can be solved.

In the following parts of this paper, we mainly focus on deep learning, how it works, and how it can be applied to develop efficient and high-performance underwater fish classifiers. We will also critically analyse the 17 DL studies found as part of our systematic literature review described earlier.

2.4 Background To Deep Learning

Deep Learning (DL) [108, 109] is a subset of ML algorithms that employs a neural network with several layers to very loosely replicate the function of the human brain by enabling it to ”learn” from huge quantities of data. The learning happens when the neural network extracts higher-level features from input training data. The term ”deep” refers

Table 2.1: List of fish classification studies using hand-crafted computer vision and accuracy between 2003-2021. *The reported accuracies were obtained from different datasets with varying settings.*

Article	Year	Classification Method	AC
An Automated Fish Species Classification and Migration Monitoring System [104]	2003	Feature vector Classification	92
Determining the appropriate feature set for fish classification tasks [102]	2005	Naive Bayes	90
Real-time underwater sorting of edible fish species [82]	2006	Naive Bayes	98
One Fish, Two Fish, Butterfish, Trumpeter: Recognizing Fish in Underwater Video [92]	2007	Support vector machine	90
Classification of guppies' (Poecilia reticulata) gender by computer vision [81]	2008	Naive Bayes	96
Automatic Fish Classification for Underwater Species Behavior Understanding [105]	2010	Discriminant Analysis Classification	92
Fish Recognition Based on Robust Features Extraction from Size and Shape Measurements Using Neural Network [97]	2010	Backpropagation	86
Fish Classification Based on Robust Features Extraction From Color Signature Using Back-Propagation Classifier [98]	2011	Backpropagation	84
Fish species classification by color, texture and multi-class support vector machine using computer vision [93]	2012	Support vector machine	97
Real-world underwater fish recognition and identification, using sparse representation [106]	2013	Sparse representation classification	81
A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage [107]	2013	Gaussian Mixture Model	97
Automatic Nile Tilapia Fish Classification Approach using Machine Learning Techniques [94]	2013	Support vector machine	94
Shape- and Texture-Based Fish Image Recognition System [99]	2013	Backpropagation	90
A General Fish Classification Methodology Using Meta-heuristic Algorithm With Back Propagation Classifier [100]	2014	Backpropagation	80
GMM improves the reject option in hierarchical classification for fish recognition [88]	2014	Support vector machine	74
Supervised and Unsupervised Feature Extraction Methods for Underwater Fish Species Recognition [78]	2014	Hierarchical Partial Classifier	93
A Feature Learning and Object Recognition Framework for Underwater Fish Images [79]	2015	Support vector machine	98
A novel tool for ground truth data generation for video-based object classification [85]	2015	K-means algorithm	93
Automated detection of rockfish in unconstrained underwater videos using Haar cascades and a new image dataset: labeled fishes in the wild [86]	2015	Haar cascade classifiers	89
Fish Classification Using Support Vector Machine [95]	2015	Support vector machine	79
Fish identification from videos captured in uncontrolled underwater environments [83]	2016	Sparse Approximated Nearest Point	94
Fish Activity Tracking and Species Identification in Underwater Video [87]	2016	Support vector machine	91
Koi Fish Classification based on HSV Color Space [80]	2016	Naive Bayes	97
Optical Fish Classification Using Statistics of Parts [101]	2016	Backpropagation	95
Shrinking Encoding with Two-Level Codebook Learning for Fine-Grained Fish Recognition [96]	2017	Support vector machine	98
Indigenous Fish Classification of Bangladesh using Hybrid Features with SVM Classifier [84]	2019	Support vector machine	94

to the usage of several layers in the neural network. Lower layers, for example in image processing, could detect edges, whereas higher layers might identify parts of the object.

2.4.1 How Deep Learning differs from Machine Learning

Machine Learning (ML) is usually referred to as a class of algorithms that can recognise patterns in data and create prediction models automatically. Deep Learning (DL) is a subclass of standard ML because it uses the same type of data and learning methods that ML applies. However, when dealing with unstructured data, e.g. text and images, ML usually goes through some pre-processing to convert it to a structured format for learning. DL, on the other hand, does not usually require the data pre-processing needed by ML. It is capable of recognising and analysing unstructured data, as well as automating feature extraction, significantly reducing the need for human knowledge (see Figure 2.2-bottom).

For example, to recognise fish in an image, ML requires that specific fish features (such as shape, colour, size, and patterns) be explicitly defined in terms of pixel patterns. This may be a challenge for non-ML specialists because it typically requires a deep grasp of the domain knowledge and good programming skills. DL techniques, on the other hand, skip this step entirely. Using general learning techniques, DL systems can automatically recognise and extract features from data. This means that we just need to tell a DL algorithm whether a fish is present in an image, and it will be able to figure out what a fish looks like given enough examples. Decomposing the data into layers with varying levels of abstraction enables the algorithm to learn complex traits defining the data, allowing for an automatic learning approach. DL algorithms may be able to determine which features (such as fishtail) are most important in differentiating one animal from another. Prior to DL, this feature hierarchy needed to be determined and created by hand by an ML expert.

2.4.2 How Deep Learning works

Deep Neural Network (DNN), also known as artificial neural network, is the basis of deep learning. DNNs use a mix of data inputs, weights, and biases to learn the data, by properly detecting, categorising, and characterising objects in a given dataset of interest. DNNs are made up of several layers of linked nodes, each of which improves and refines the network prediction or categorisation capabilities. For instance, Fig. 2.3 shows a popular DNN architecture for image processing, called UNET [32]. UNET, which is a fairly complex deep learning architecture, is composed of a few different components and layers, to achieve a specific learning goal, i.e. to segment fish body in an input image.

Any DNN is composed of three types of layers, namely input, output, and hidden layers. The visible layers are the input and output layers (see Figure 2.6). The DL model gets the data for processing in the input layer, and the final prediction or classification is generated in the output layer. In a typical neural network, including a DNN, the learning happens through two general processes, *i.e.* forward and backward propagations. Forward propagation refers to the propagation of input data through the network layers to generate a prediction or classification result. Backward propagation or, backpropagation in short, is where the learning happens in the network. Backpropagation uses a training model that determines prediction errors and then changes the weights and biases of the neural network by going backwards through its layers. Forward propagation and backpropagation work together to allow a neural network to generate predictions and reduce the network errors. Through many iterations of backward and forward propagation, the neural network prediction or classification accuracy improves.

Almost all DNNs work on and through the same principles described above. However, different DL networks and architectures are used to solve different tasks. For instance, CNNs, which are commonly used in computer vision and image classification applications, can recognise characteristics and patterns within an image, allowing tasks such as object detection and recognition to be accomplished. However, in tasks with a different nature, such as natural language processing, speech recognition, or timeseries forecasting [110], Recurrent Neural Networks (RNNs) are commonly employed. Despite the differences in their architectures, many DL techniques, use the concept of supervised learning to process their input data and accomplish different tasks.

2.4.3 Supervised Learning

Supervised learning is a method used to enable finding and optimising a function that maps an input to its corresponding output in an input-output object pair, also known as training example [111]. Supervised learning uses a set of training examples based on manually-labelled training data prepared by human observers or 'supervisors', hence the name for the learning method.

The aim of supervised learning is to generate an inferred function, f, that maps to the training examples, and can then be used to map to new examples outside of the training examples. In order to accomplish any general task, a computer can be programmed to find function f to map X to Y, *i.e.* $(f : X \mapsto Y)$, where X is an input domain and Y is an output domain. For example, in an image classification task, X is the dataset of images and Y is a set of corresponding classification labels, which determine whether an object

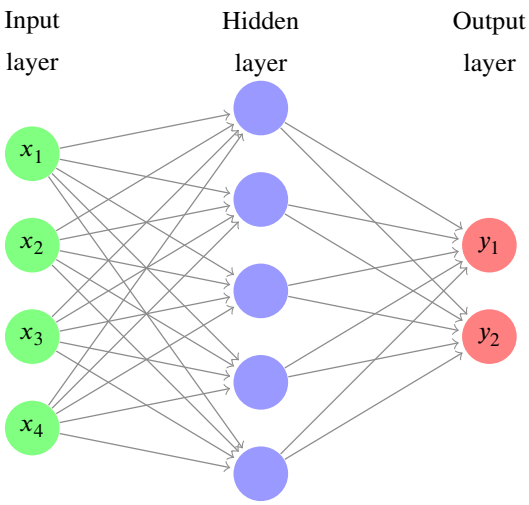

Figure 2.6: A diagram of a single-layer neural network, composed of input, hidden, and output layers.

is present in the respective image in the dataset or not.

To determine the function f that can recognise, for instance, a fish in an image using DL, one solution is to do feature engineering. However, it is usually very difficult to perform this, *i.e.* hand-pick features of the fish, based on the domain knowledge that comes from the training dataset. In addition, most of the time, the hand-picked features need to be pruned to reduce their pixel dimensionality. Comparatively, it is often more feasible to collect a large dataset of $(x, y) \in X \times Y$ to find the mapping function f, and this affords supervised learning advantage as an alternative mapping technique compared with direct feature engineering. Specifically, in the fish classification task, a large dataset of fish images is collected, where each image x is labelled with y that shows the presence or absence of a fish, without the need to hand-pick its features.

One of the main supervised learning approaches is training a neural network, which is the foundation of deep learning, especially for computer vision applications such as fish image processing. We, therefore, dedicate the next subsection to neural networks and their underlying working principles.

2.4.4 Neural Networks

A 'neural network' [112] is a computer program originally conceived by mimicking actual cerebral neural networks that make up the brain's grey matter. A computer's neural network, a.k.a. an artificial neural network, "learns" to do a specific task by using a large amount of data, usually through supervised network training that does not involve

any task-specific rules. As briefly mentioned, a neural network is constructed from three types of layers: an input layer, hidden or latent layers, and an output layer (see Figure 2.6). These layers include processing neurons within them (coloured circles in Figure 2.6), and connecting synapses (weights) between them (edges in the figure).

The input layer is the gate to the network. It provides information to the network from outside data, and no calculation is made in this layer. Instead, input nodes pass the information on to the hidden layer. This layer is not visible to the outside world and serves as an abstraction of the inputs, independent of the neural network structure. The hidden layer (layers) processes the data received from the input layer and transfers the results to the output layer. Finally, the output layer brings the information that the network has learned into the outside world.

Learning in a neural network happens through minimising a loss function. Generally, a loss function is a function that returns a scalar value to represent how well the network performs a specific task. For example, in image classification, the network is expected to correctly classify all the images containing a fish as fish, and all those not including a fish, as no fish, returning a loss value of zero. During learning, the network receives a large amount of input data, e.g. thousands of fish images, and eventually learns to minimise the loss between its predicted output and the true target value. In the case of supervised learning, these true target values are provided to the network, to find function f described in the previous section, to minimise the loss function. This minimisation happens through optimising f using an algorithm such as Stochastic Gradient Descent (SGD) [113] that helps find network weights/parameters that minimise the loss.

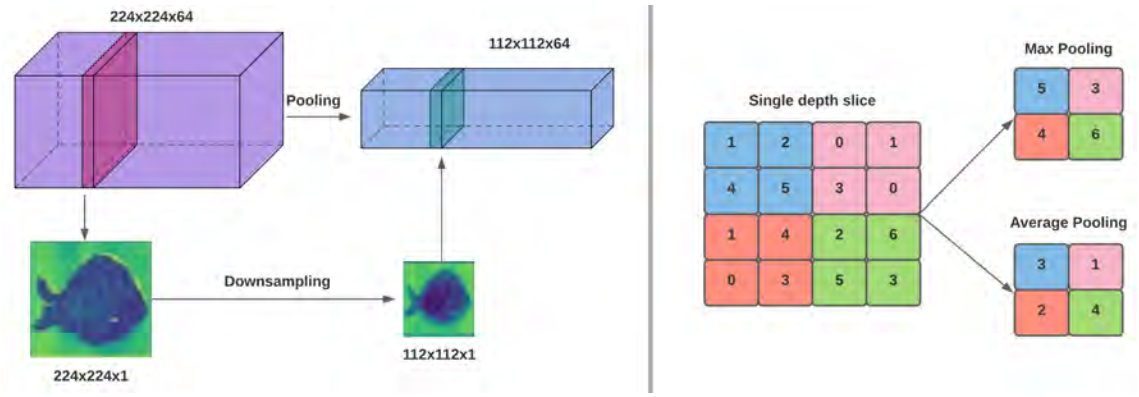

Figure 2.7: Schematic diagram of pooling layer: (**Left**) single feature map spatially downsampled from a representation block with shape $224 \times 224 \times 1$ to a new representation of shape $112 \times 112 \times 1$. (**Right**) types of pooling layer (max-pooling and average-pooling).

2.4.5 Convolutional Neural Network

CNNs are probably the most commonly used artificial neural networks. They have been the dominant deep learning tool in computer vision and have been widely used in underwater marine habitat monitoring [24]. CNNs are broadly designed after the neuronal architecture of the human cortex but on much smaller scales [114]. A CNN [115] is specifically designed for dealing with datasets that have some spatial or topological features (e.g. images, videos), where each of the neurons are placed in such a manner that they overlap and thus react to multiple spots in the visual field. A CNN neuron is a simple mathematical design of the human brain's neuron that is utilised to transform nonlinear relationships between inputs and outputs in parallel. There are two primary layer types in a CNN, i.e. convolutional layers and pooling layers, which generate feature maps, as explained in the following subsections.

Convolutional Layer

In this layer, the convolutional processes (*i.e.*, the multiplication of a small matrix of the input neurons by a small array of weights called filter) are used on limited fields (which depend on the size of the filter) to avoid the need to learn billions of weights (parameters), which would be required if all the neurons in one layer are connected to all the neurons in the next layer. This excessive computation is avoided through the weight-sharing of convolutional layers combined with filters for their corresponding feature maps. In a convolution operation, a small matrix of the input neurons is multiplied in its same-sized matrix, called a filter. In a convolutional layer, this convolution operation happens by sliding the filter on the entire input neurons, generating a feature map. Filters work on a reduced area of the input (convolutional kernel). Convolutional layers can either use the same kernel size or they can use different kernel sizes, which makes it possible to extract complex features from the input using fewer parameters. In addition, weight-sharing is useful in avoiding model overfitting, i.e. memorising the training data, [116], while also reducing computing memory requirements and enhancing learning performance [117].

Pooling Layer

This layer is used to reduce the spatial dimension (not depth) of the input features and add control for avoiding overfitting by reducing the number of representations with a specified spatial size. Pooling operations can be done in two different ways, i.e. Max and Average pooling. In both methods (see Figure 2.7), an input image is down-scaled in

size, by taking the maximum of 4 pixels and down-sampling them to one pixel. Pooling layers are systematically implemented between convolutional layers in conventional CNN architectures. The pooling layers work on each channel (activation map) individually and downsample them spatially. By having fewer spatial information, pooling layers make a CNN more computationally efficient.

Feature Maps

Feature Maps, also called Activation Maps, are the result of applying convolutional filters or feature detectors to the preceding layer image. The filters are moved on the preceding layer by a specified number of pixels. For instance, in Figure 2.8, there are 37 filters of the size 3×3 that move across the input image with a stride of 1 and result in 37 feature maps.

The majority of CNN layers are convolutional layers. These layers are used to apply the same convolutional filtering operation to different parts of the image, creating "neurons" that can then be used to detect features, like the edges and corners. A collection of weights connects each neuron in a convolutional layer to the preceding layer's feature maps, or to the input layer image. The feature maps help visualise the features that the CNN is learning to give an understanding of the network learning process, as shown in Figure 2.8.

2.5 Applications of Deep Learning in Fish-Habitat Monitoring

In a recent special issue titled "Applications of machine learning and artificial intelligence in marine science" published in the International Council for the Exploration of the Sea (ICES) journal of marine science [118], many uses of deep learning and CNNs have been shown. These include identifying the species of harvested fish [119], analysis of fisheries surveillance videos [120], and natural mortality estimation [121]. Other published works have used CNN for other marine applications such as automatic vessel detection [122], and analysis of deep-sea mineral exploration [123]. However, in this paper we focus on using CNNs for CV tasks.

These tasks are mainly designed to extract knowledge from underwater videos and images. Despite the recent use of CNNs for various visual analysis tasks such as segmentation [28, 124–126], localisation [127–129], and counting [10, 130, 131], the most

common and the widest studied CV task in underwater fish habitat monitoring has been classification. Therefore, in this paper, we focus mainly on classification of underwater fish images. We survey some of the latest works on fish classification and provide a high-level technical discussion of these works.

The task of classification is defined as classifying the input samples into different categories, usually based on the presence or absence of a certain object/class, in binary classification; or the presence of several different objects belonging to different classes, in multi-class classification [132]. Similarly, image classification is concerned with assigning a label to a whole image based on the objects in that image. Conceivably, an image can be labelled as fish, when there is a fish present in it, or negative when no fish is present. Similarly, images of different species should be automatically assigned to their respective classes or given a label representing their class.

Classification is a difficult process if done manually, because an image may need to be categorised into more than one class. In addition, there may be thousands of images to be classified, which makes the task very time-consuming and prone to human error. Consequently, automation can help perform classification quicker and more efficiently.

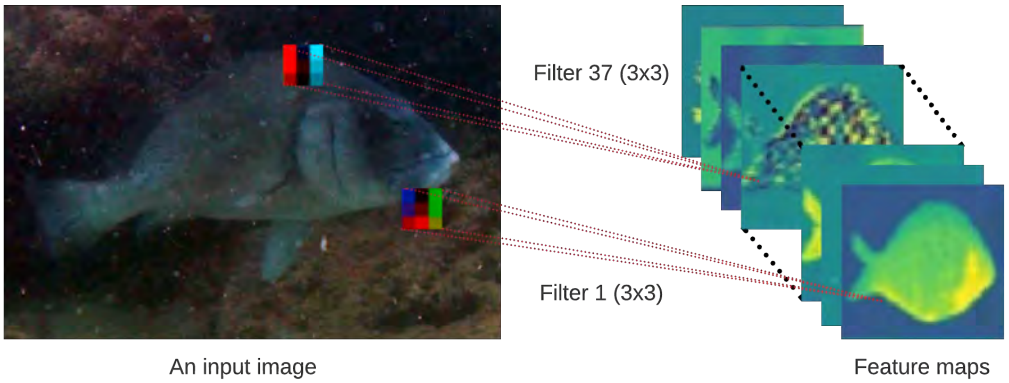

Figure 2.8: Schematic diagram of feature maps of the CNN used in the classification task. The feature map is a two-dimensional representation of an input image. Here (3×3) is the size of the filter slid over the entire image to generate feature maps.

In the context of fish and marine habitat monitoring, CV offers a low-cost, long-term, and non-destructive observation opportunity. One of the initial tasks performed using deep learning on CV-collected marine habitat images is fish classification, which is a key component of any intelligent fish monitoring systems, because it may activate further processing on the fish image. However, underwater monitoring based on image and video processing pose numerous challenges related to the hostile condition under which the fish

images are collected. These include poor underwater image quality due to low light and water turbidity, which result in low resolution and contrast. Additionally, fish movements in an uncontrolled environment can create distortion, deformations, occlusion, and overlapping. Many previous works [133–135] have tried to address these challenges. Some of these works focused on devising new methods to properly extract traditional low-level features such as colours and textures using mean shift algorithm [136], in the presence of the challenges. However, these works have not been very successful compared to DL approaches.

With the inception of CNNs, many researchers utilised them to extract both high-level and low-level features of input images. These features, which can be automatically detected by the CNN, carry extensive semantic information that can be applied to recognise objects in an image. In addition, CNNs have the ability to address the challenges outlined above. Therefore, they are currently the main underwater image processing tool in literature for fish classification, as shown in Tables 2.2 and 2.3. These tables list some of the latest classification works, while providing details about the DL models used and the framework within which the model was implemented. It also provides information about the data source, as well as the pre-processing of the data and its labels, while reporting the Classification Accuracy (CA) and a short comparison with other methods if the reviewed work has provided it. One of the main metrics when comparing different methods for classification is their CA, which is defined as the percentage of correct predictions by the network.

$$CA = (TP + TN)/(TP + TN + FP + FN), \tag{2.1}$$

where TP (True Positive) and TN (True Negative) represent the number of correctly classified instances, while FP (False Positive) and FN (False Negative) represent the number of incorrectly classified instances. For multi-class classification, CA is averaged among all the classes.

DL algorithms are gaining momentum in their growing accuracy in different applications. However, they have inherent limitations, which should be considered before choosing a DL algorithm for a given application. This is because accuracy, for example in a fish classification task, may significantly differ from true accuracy due to the distribution of samples in the training and testing populations. To address this limitation of classification accuracy, the Receiver Operating Characteristics (ROC) [137] and Area Under The Curve (AUC) [138] are widely used as a standard measure for determining the performance of a model in a binary classification setting. Their definition is very similar to accuracy but they help one understand the probability that the classifier produces correct outputs with

desired levels of true positives and false negatives, using a certain classification threshold.

The works in Tables 2.2 and 2.3 can be divided into two general categories. The first category deals with designing effective CNNs that address the challenge of unconstrained, complex, and noisy underwater scenes, while the second category also tries to address the usual problem of limited fish training datasets.

As mentioned, when processing unconstrained underwater scenes specific attention should be paid to implementing a classification approach that is capable of handling variations in light intensity, fish orientation, and background environments, and similarity in shape and patterns among fish of various species. In order to overcome these challenge and to improve classification accuracy, various works have devised different methodologies. In [139], the authors used different activation functions to examine the most suitable for fish classification, while in [67] different number of convolutional layers and different filter sizes were examined. In [68], the authors used a CNN model in a hierarchical feature combination setup to learn species-dependent visual features for better accuracy. In another work [69], principal-component analysis was used in two convolutional layers, followed by binary hashing in the non-linear layer and block-wise histograms in the feature pooling layer. Furthermore, a single-image super-resolution method was used in [70] to resolve the problem of limited discriminative information of low-resolution images. Moreover, [140] used two independent classification branches, with the first branch aiming to handle the variation of pose and scale of fish and extract discriminative features, and the second branch making use of context information to accurately infer the type of fish. The reviewed works show that depending on the type of environment and fish species similarities in the dataset under consideration, various techniques should be considered and investigated to find the best classification accuracy.

As already mentioned, data gathering in the wild is sometimes very difficult and challenging, thus to maximize the success rate of training, it is essential to consider gathering field data from the beginning of the project. This ensures that the collected training dataset has good sample diversity including samples collected at different environmental conditions such as water turbidity and salinity, and it captures fish species similarities. Diversity and comprehensiveness in the dataset is one of the key factors in reaching high classification accuracies when the model is deployed in the real world. Data augmentation is another important method that can help improve the classification accuracy, through increasing the dataset size and diversity. An alternative to data augmentation is transfer learning, but the model should be always fine-tuned to the new dataset to maximize

accuracy. Image pre-processing is another important technique that can help improve classification accuracy, and should be considered when working with new fish datasets.

Dataset limitation, i.e. having limited number of fish images from different species, and/or having few numbers of different fish etc, is another challenge in underwater fish habitat monitoring in general and in fish classification, in specific. This challenge has been addressed in [24, 141–143] using transfer learning.

Transfer learning is a ML method that works by transferring information obtained while learning one problem or domain to a different but related problem or domain. Comparing a randomly initialised classifier with another one pre-trained on ImageNet [144], Saleh *et al.* [24] achieved a fish classification accuracy of 99%, outperforming the randomly-initialised classifier, significantly. This finding shows that transfer learning can bring learned information from the ImageNet learning domain to fish classification domain and can be a useful and crucial method for evaluating fish environments. Transfer learning was also used in [145] where general-domain above-water fish image learning was transfered and used for underwater fish classification. In the same way, to train large-scale models that are able to generate reasonable results, [146] collected 1000 fish categories with 54,459 unconstrained images from various professional fish websites and Google engine.

In addition to transfer learning, some works have developed specific machine learning techniques suiting their applications. For instance, in a previous study [147], a pre-trained CNN was used as a generalised feature extractor to avoid the need for a large amount of training data. The authors showed that by feeding the CNN-extracted features to a Support Vector Machine (SVM) classifier [148], a CA of 94.3% for fish species classification can be achieved, which significantly outperforms a stand-alone CNN achieving an accuracy of 53.5%. Also, [149] used the same techniques in [147] to achieve a CA of 98.79%. In addition, [150] developed a new technique for fish classification by modifying AlexNet [103] model with fewer number of layers. Moreover, [42] presented a labelling efficient method of training a CNN-based fish-detector on a small dataset by adding 27,000 above-water and underwater fish images.

CNNs are sometimes capable of surpassing human performance in identifying fish in underwater images. By training a CNN on $900,000$ images, Villon *et al.* [151] could achieve a CA of 94.9% while human CA was only 89.3%. This result was achieved mainly because the CNN was able to successfully distinguish fish that were partially occluded by corals or other fish, while human could not. Furthermore, the best CNN model developed in [151] takes 0.06 seconds on average to identify each fish using typical hardware (Titan

X GPU). This demonstrates that DL techniques can conduct accurate fish classification on underwater images cost-effectively and efficiently. This facilitates monitoring underwater fish and can advance marine studies concerned with fish ecology.

If DL methods are going to be deployed widely for different marine applications such as fish classification, there is a need to implement them efficiently, so that they can run on low-power embedded systems, which can run in real-time on mobile devices such as underwater drones. To that end, Meng *et al.* [152] have developed an underwater drone with a panoramic camera for recognising fish species in a natural lake to help protect the environment. They have trained an efficient CNN for fish recognition and achieved 87% accuracy while requiring only 6 seconds to identify 115 images. This promising result shows that, DL can be used to classify underwater fish while also satisfying the real-time conditions of mobile monitoring devices. In addition, other efficient hardware design approaches that have proven useful in reducing power consumption and increasing speed in classification task in other domains such as agriculture [153] can be adopted on edge underwater processors.

In DL applications, video storage is currently a bottleneck that may be bypassed with real-time algorithms, because they only need to store some and not all the video frames in memory and process them in-situ, as they become available. This eliminates the time it takes for all the frames to be stored and retrieved from memory. This is helpful in situations where large amounts of data have to be processed quickly, for example, in an underwater fish observation camera, where frames are collected continuously and should either be stored locally or transfered to surface, which are both costly and mostly impossible. Using real-time processing algorithms, the frames are processed and only the information obtained, *i.e.* the number of fish in a frame are sent or stored, which is much lighter than the entire frame.

2.6 Challenges and Approaches to Address Applications of DL

Despite the rapid improvement of DL for marine habitat monitoring through visual analysis, four main challenges still exist [53]. The first challenge is to develop models that can generalise their learning and perform well on new unseen data samples. The second challenge is limited datasets available for general DL tasks, and in particular for marine visual processing tasks. The third challenge is lower image quality in underwater scenarios. The

Table 2.2: Summary of recent DL research works performing the task of fish classification

Article	DL Model	Framework	Data	Annotation/Pre-processing/Augmentation	Classes and Labels	Perf. Metric	Metric Value	Comparisons with other methods
Recognition of Fish Categories Using Deep Learning Technique [139]	CNN	Keras, Tensorflow	Authors-created dataset containing 560 fish images, 400 training and 160 test images.	Each image is assigned the fish species name as a label	10 classes of 10 different fish species	CA	95%	NA
Comparison of Different DL Structures for Fish Classification [67]	CNN	Torch	The public QUT fish dataset contains 3960 images of 468 fish species in different environments.	Each image is assigned the fish species name as a label	468 classes of 468 different fish species	CA	46.02%	NA
Fish Species Classification in Unconstrained Underwater Environments Based on DL [68]	CNN	NA	The images are from the public Fish4Knowledge dataset (LifeCLEF 2014, LifeCLEF 2015)	Each image is assigned the fish species name as a label	25 classes of 25 different fish species	CA	96.75%	Comparison with the conventional SVM machine learning tool that achieved 83.94%
Deep-Fish: Accurate Underwater Live Fish Recognition with a DL Architecture [69]	CNN	Matlab	The images are from the public Fish4Knowledge dataset	Each image is assigned the fish species name as a label	23 classes of 23 Different fish species	CA	98.64%	Comparison with conventional machine learning tools as baseline methods achieving 93.58%
Fish Recognition from Low-resolution Underwater Images [70]	CNN	NA	93 videos from LifeCLEF 2015 fish dataset	Each image was annotated by drawing a bounding box and labelling by species name	15 classes of 15 different fish species	CA	76.57%	Authors used the traditional gabor features and dense sift features that generated CA of 38.28% and 28.63%, respectively.
Automatic Fish Classification System Using DL [140]	CNN	NA	Eight target categories: Albacore tuna, Bigeye tuna, Yellowfin tuna, Mahi Mahi, Opah, Sharks, Other.	Each image is assigned the fish species name as a label	8 classes of 8 different fish species	CE	0.578, 1.387	Ranked 17th on Kaggle leaderboard on test set at stage 1 and 16th at stage 2.
A Realistic Fish-habitat Dataset to Evaluate DL Algorithms For Underwater Visual Analysis [24]	ResNet-50 CNN	Pytorch	Authors-created database containing 39,766 images from 20 habitats in remote coastal marine environments of tropical Australia	point-level and semantic segmentation labels	20 classes of 20 different fish species	CA	0.99	NA
Deep Learning for Underwater Image Recognition in Small Sample Size Situations [141]	CNN	Caffe	The images are from the public Fish4Knowledge dataset	Each image is assigned the fish species name as a label	10 classes of 10 different fish species	CA	85.08%	NA
Underwater Fish Species Classification using CNN and DL [142]	CNN	NA	27000 images from the public Fish4Knowledge dataset	Each image is assigned the fish species name as a label	23 fish classes	CA	96.29%	NA

Table 2.3: Summary of recent DL research works performing the task of fish classification

Article	DL Model	Framework	Data	Annotation/Pre-processing/Augmentation	Classes and Labels	Perf. Metric	Metric Value	Comparisons with other methods
Underwater Live Fish Recognition by Deep Learning [143]	AlexNet CNN	Matlab	27000 images from the public Fish4Knowledge dataset	Each image is assigned the fish species name as a label	23 classes of 23 different fish species	CA	99.45%	NA
WildFish++: A Comprehensive Fish Benchmark for Multimedia Research [146]	CNN	NA	Authors-created dataset of 54,459 labelled images from various professional websites and Google engine	Each image is assigned the fish species name as a label	100 classes of 1000 different fish species	CA	74.7%	Comparison with other state-of-the-art approaches
Automatic Fish Species Classification in Underwater Videos: Exploiting Pre-trained DNN Models to Compensate for Limited Labelled Data [147]	CNN	MATLAB	The dataset contains 50 to 120 10-second video clips of 16 species from Western Australia during 2011 to 2013.	Each image is assigned the fish species name as a label	16 classes of 16 Different fish species	CA	89.0%	Comparison of their proposed method of CNN+SVM achieving a CA of 89.0% with two previous works; SRC (Hsiao *et al.* [154]) 49.1% and CNN (Salman *et al.* [68]) 53.5%
Underwater Fish Species Recognition using Deep Learning Techniques [149]	CNN	Keras, Tensorflow	35000 images from the public Fish4Knowledge dataset	Each image is assigned the fish species name as a label	23 classes of 23 different fish species	CA	98.79%	NA
Automatic Fish Species Classification Using Deep Convolutional Neural Networks [150]	modified AlexNet CNN	Tensorflow	The images are from two public datasets: QUT fish dataset and LifeClef-15	Each image is assigned the fish species name as a label	6 classes of 6 different fish species	CA	90.48%	Comparing their proposed modified AlexNet achieving a CA of 90.48% with original AlexNet CA of 86.65%
Underwater Fish Detection with Weak Multi-Domain Supervision [42]	CNN	Keras, Tensorflow	Authors-created dataset of 40000 labelled fish images from video sequences	Each image is labelled as Fish or no fish	2 classes	CA	99.94%	NA
A Deep Learning Method for Accurate and Fast Identification of Coral Reef Fishes in Underwater Images [151]	GoogLeNet CNN	Caffe	Authors-created dataset containing 450,000 images from over 50 reef sites around the Mayotte island	Annotation included drawing a rectangle around a single fish and associating the species name as label.	20 classes of 20 different fish species	CA	94.9%	Comparing accuracy to human experts. The rate of correct identification was 94.9%, greater than the rate of correct identification by humans (89.3%).
Underwater-Drone With Panoramic Camera for Automatic Fish Recognition Based on Deep Learning [152]	CNN	NA	Authors-created dataset of 100 labelled images from Google search engine	Each image is assigned the fish species name as a label	4 classes of 4 different fish species.	CA	87%	NA

fourth challenge is the gap between DL and ecology.

To address these challenges, various computer algorithms and techniques have been developed. In the following subsections, we explain the challenges in detail and briefly review various approaches to address them. However, we do not intend to include details of these approaches as they are out of the scope of this paper. The interested reader is invited to refer to relevant DL materials and the cited papers.

2.6.1 Model Generalisation

One of the most difficult challenges in DL is to improve deep convolutional networks generalisation abilities. This refers to the gap between a model's performance on previously observed data (*i.e.* training data) and data it has never seen before (*i.e.* testing data). A wide gap between the training and validation accuracy is usually a sign of overfitting. Overfitting occurs when the model accurately predicts the training data, mostly because it has memorised the training data instead of learning their features.

One way to monitor overfitting is by plotting the training and validation accuracy at each epoch during training. That way, we will see that if the gap between the validation and training acuuracy/error is widening (over- or under-fitting) or narrowing (learning). A well-known and effective method for improving the generalisability of a DL model is to use regularisation [155]. Some of the regularisation methods applied to fish and marine habitat monitoring domains include transfer learning [156], batch normalisation [28], dropout [150], and using a regularisation term [130].

2.6.2 Dataset Limitation

Another challenge of training DL models is the limited dataset. DL models require enormous datasets for training. Unfortunately, most datasets are large, expensive, and time-consuming to build. For this reason, model training is usually conducted by collecting samples from a small number of datasets, rather than from a large number of datasets.

A dataset can be categorised into two parts: labelled data and unlabeled data. The labelled data is the set of data that needs the labelling of classes, e.g. fish species in an image, or absence or presence of fish in an image. The unlabeled data is the set of data that has not been processed. The labelled data forms the training set whose size is closely related to the accuracy of the trained model. The larger the training set, the more accurate the trained model. Large training set, however, are expensive to build. They require a large number of resources, such as people-hours, space, and money, making it

very difficult for many researchers to achieve them, and in turn hinders their research.

Since it is difficult to obtain a large labelled dataset, various techniques have been proposed to address this challenge. Some of the techniques applied to the fish and marine habitat monitoring domains include transfer learning [157], data augmentation [24, 67], using hybrid features [158–160], weakly supervised learning [9], and active learning [161].

2.6.3 Image Quality

Underwater image recognition's average accuracy lags significantly behind that of terrestrial image recognition. This is mostly owing to the low quality of underwater photos, which frequently exhibit blurring, and colour deterioration, caused by the physical characteristics of the water and the hostile underwater environment.

Most CV applications perform some initial preprocessing of images before feeding them to their image processor. In underwater scenarios, these preprocessing techniques are typically used to enhance the image quality. Preprocessing can also help with the red channel information loss problem, which is required for obtaining relevant colour data. The red channel information loss problem is about losing the actual intensity of the red colour in the scene, for instance, compared to the blue and green colour channels. This is more pronounced in the underwater environment and as the depth increases, which attenuates red channel values more strongly than the other colour channels. We should, therefore, consider that the red channel value depends not only on the distance from the subject but also on the intensity of the light reflected by the subject, as the reflection of intense light is typically much stronger than that of a light of a very low intensity. Another issue that arises in the detection of a specific target in an underwater image is the fact that multiple pixels can potentially be activated in the image in theform of an object. For example, sunlight shining through a periscope lens can cause spurious activation of a given pixel. There is a need for a reliable method and system for determining whether a given pixel in a remote underwater image is activated by some cause other than the presence of a target in the area of the image.

Preprocessing of underwater photos has been extensively researched, and several solutions have been devised for correcting typical underwater image artefacts [162, 163]. However, the image quality produced by these approaches is subjective to the observer, and because acquisition settings vary so widely, these methods may not be applicable to all datasets. According to empirical results [164, 165], the current tendency appears to be to perform picture repair and enhancement processes based on the dataset, i.e. determin-

ing the most appropriate preprocessing strategy for a specific dataset. This strategy also depends on the purpose (e.g labelling, classification or both) of the images in the dataset.

In addition, basic image enhancement techniques have been shown to be effective in improving image quality. For instance, in [159] increasing the uniformity of the background was used to boost picture contrast in underwater images for marine animal classification. This is a strong indicator that simple enhancing approaches might result in increased performance. Furthermore, some recent studies have employed DL algorithms to enhance image quality using low-quality images. In [166], for example, end-to-end mapping is performed between low-resolution and high-resolution images.

When compared to state-of-the-art handcrafted and traditional image enhancement methods, DL-based algorithms typically perform better in addressing picture quality in terrestrial photos. However, significant new research is required to customise these DL-based techniques for underwater images and maritime datasets. This poses as a future research opportunity for image quality enhancement in fish monitoring applications. Below, we discuss some more opportunities.

2.6.4 Deep Learning Gap

DL is an emerging field that has a lot to offer in terms of ecology. The first and most obvious ecological applications are fish classification or fish count. However, there is still a gap between the DL-predicted fish counts and, for example, absolute abundance (fish per area or volume unit). The existing DL literature discusses mainly the use of CNNs for the ecological problems of species classification or fish counting. However, the absolute abundance of fish is important for ecological research and species conservation.

Another important problem in ecological research is fish population dynamics. A step in addressing this problem is to analyze long-term data on fish movements and fish densities. However, such long-term datasets are relatively rare and expensive to obtain. Hence, there is a need to obtain as much information as possible from the small amount of data given. This requires novel methods to give an accurate long-term estimate of fish densities or, even better, an estimate of the absolute abundance of fish.

Other exemplar ecological questions that can be addressed using DL include species habitat selection, or the relationship between the physical environment and the life history of species [167–169]. DL methods can help us with this because they can take advantage of all the available information. The current state of DL research can be improved by considering alternative network architectures, more complex training algorithms, and more detailed knowledge of the problem domain. The existing DL literature suggests that we

may see many new methods in the future. Most of them still do not have sufficient data to prove that they can outperform existing methods. There are, however, examples of successful applications, such as fish classification. For many ecological problems, a DL method can give very accurate predictions of fish densities or absolute abundance. However, it remains unclear whether this accuracy can be obtained only with the appropriate method or whether this is a property of the particular dataset on which the method was trained. From this perspective, the development of a general method for predicting fish densities and absolute abundance from very little data is a major problem in ecology.

One potential approach to solving this problem is to take advantage of DL models trained on other datasets, as long as they are related to the fish density/abundance problem. The ecological literature suggests that the relationship between the physical environment and the life history of species (e.g., fish density) is likely to be complex because the physical environment differs from species to species. Therefore, we may be able to find many similar datasets on other related problems (e.g., environmental science or engineering). In addition to developing and testing general methods to estimate the absolute abundance of fish from very little data, there is a need to develop general methods that can take advantage of the ecological knowledge and domain-specific data from a particular problem.

2.7 Opportunities in Application of DL to Fish Habitat Monitoring

New methods and techniques will need to be devised to improve the accuracy of deep learning models for various marine habitat monitoring applications and to bring them closer to their terrestrial counterparts.

2.7.1 Spatio-temporal and Image Data Fusion

Most of the current marine habitat monitoring and visual processing tools only use image-based data to train their model to understand the habitats and monitor the environment. In such tools, each frame or image is separately processed and spatiotemporal correlations across neighbouring frames are simply overlooked. Exploiting this extra information and fusing it with the image-processing model can be beneficial [36]. For instance, fusing a master-slave camera setup with LSTM [170] can help to learn the kinematic model of fish in a 3D fish tracking system. Future works should consider including spatiotemporal

information in training their model and understanding the scene. In particular, approaches similar to Long short-term memory (LSTM) networks or other RNN models can be used in conjunction with CNNs, to obtain improved classification or prediction outcomes by taking advantage of the time-domain information. For example, An RNN and a CNN model are combined in [171] to achieve better performance for salmon feeding action recognition from underwater videos. In [172], the authors propose a spatio-temporal recurrent network to classify behavioural patterns. Similar schemes have been proposed in [173]. However, their performance and complexity heavily rely on the ability of the RNN to track the temporal relations of the frames and on the effectiveness of the CNN.

For instance, estimating and monitoring fish development based on previous continuous observations, and analysing fish behaviour are some of the applications where time domain information will be not only useful but also critical. Such models can also be used to build novel video-based protocols for the surveillance of critically endangered reef fish biodiversity.

2.7.2 Underwater Embedded and Edge Processing

DNNs have proven to be successful in both industry and research in recent years, particularly for CV tasks. Specifically, large-scale DL models have had a lot of success in real-world scenarios with large-scale data. This is mainly due to their capacity to encode vast amounts of data and handle millions of model parameters that enhance generalisation performance when new data is evaluated. However, this high computational complexity and substantial storage requirement makes them difficult to use in real-time applications, especially on devices with restricted resources (e.g. embedded devices and underwater edge processors for online monitoring). One approach to address this is to use compressed networks such as binarised neural networks, which have shown promise toward reaching low-power and high-speed edge inference engines [153], for near-underwater-sensor processing. This can significantly improve underwater image analysis capabilities, because the collected large-volume images do not need to be transferred to surface for processing, and only the low-volume results can be communicated to shore. This also solves another problem, which is the challenging underwater communication [174].

2.7.3 Combining Data from Multiple Platforms

The use of different data collection platforms such as autonomous underwater vehicles (AUVs) or occupied submarines, can provide different image data from different per-

spectives of the same or different underwater habitats, to train more effective DNNs. In addition, using simultaneous data from multiple platforms can give more monitoring information, for instance, of fish distribution patterns, especially in situations where the number of platforms is limited. However, combining data from multiple platforms introduces some challenges such as the lack of ground truth (e.g., the number of fish in the sampled area for all the platforms), and the need to develop techniques that can integrate these data in a robust manner. Future research can work toward addressing these challenges to exploit the significant benefits of multiple platform data combination.

2.7.4 Automated Fish Measurement and Monitoring

DL can be used to achieve automated fish measurements, which may be useful in underwater fish monitoring, for instance to survey fish growth [36] through monitoring of fish length [175] and abundance [176]. Here, abundance means the number of fish in an image or video frame, and not the fish count per area or volume unit. In addition, automated measurements can realise remote fish assessments, for example when the monitoring locations are remote, or the environmental conditions and or potential hazards do not allow frequent underwater scouting by human.

DL can also be used for automation of monitoring of other fish biological variables such as their movement dynamics, present species, and their abundance and biomass. On top of these, DL can be used to automate understanding of environmental and habitat features. To achieve these, new datasets should be collected, and new or existing DL techniques should be devised or customised in future research.

2.8 Conclusion

Deep Learning (DL) sits at the forefront of the machine learning technologies providing the processing power needed to enable underwater video to fulfill its promise as a critical tool for visual sampling of fish. It offers efficient and accurate solutions to the challenges of adverse water conditions, high similarity between fish species, cluttered backgrounds, occlusions among fish, that have limited the spatio-temporal consistency of underwater video quality. As a result, DL, complemented by many other advances in monitoring hardware and underwater communication technologies, opens the way for underwater video to provide comprehensive fish sampling. This can span from shallow fresh and marine waters to the deep ocean, opening the way for the development of the truly com-

parative understanding of marine and aquatic fish fauna and ecosystems that has hitherto been impossible. At least as importantly, DL solves the problem of handling the vast quantities of data produced by underwater video in a consistent and cost-effective way, converting a prohibitively expensive activity into a simple issue of computer processing. By enabling the processing of vast quantities of data, DL allows underwater fish video surveys to be conducted with unprecedented levels of spatial and temporal replication enabling the massive knowledge advances that flow from the ability of underwater videos to be deployed contemporaneously across many habitats, and at many spatial scales, or to provide continuous data over time.

DL, and associated techniques, have the potential for widespread use in marine habitat monitoring for (1) data classification and feature extraction to improve the quality of automatic monitoring tools; or (2) to provide a reliable means of surveying fish habitats and understanding their movement dynamics. While this will allow marine ecosystem researchers and practitioners to increase the efficiency of their monitoring efforts, effective development of DL will require concentrated and coordinated data collection, model development, and model deployment efforts, as well as transparent and reproducible research data and tools, which help us reach our target sooner.

Chapter 3

Applications of Deep Learning in Fish Habitat Monitoring: A Tutorial and Survey

Marine ecosystems and their fish habitats are becoming increasingly important due to their integral role in providing a valuable food source and conservation outcomes. Due to their remote and difficult to access nature, marine environments and fish habitats are often monitored using underwater cameras to record videos and images for understanding fish life and ecology, as well as for preserve the environment. There are currently many permanent underwater camera systems deployed at different places around the globe. In addition, there exists numerous studies that use temporary cameras to survey fish habitats. These cameras generate a massive volume of digital data, which cannot be efficiently analysed by current manual processing methods, which involve a human observer. DL is a cutting-edge AI technology that has demonstrated unprecedented performance in analysing visual data. Despite its application to a myriad of domains, its use in underwater fish habitat monitoring remains under explored. In this Chapter, we provide a tutorial that covers the key concepts of DL, which help the reader grasp a high-level understanding of how DL works. The tutorial also explains a step-by-step procedure on how DL algorithms should be developed for challenging applications such as underwater fish monitoring. In addition, we provide a comprehensive survey of key deep learning techniques for fish habitat monitoring including classification, counting, localisation, and segmentation. Furthermore, we survey publicly available underwater fish datasets, and compare various DL techniques in the underwater fish monitoring domains. We also discuss some challenges and opportunities in the emerging field of deep learning for fish habitat processing. This Chapter is written to serve as a tutorial for marine scientists who

would like to grasp a high-level understanding of DL, develop it for their applications by following our step-by-step tutorial, and see how it is evolving to facilitate their research efforts. At the same time, it is suitable for computer scientists who would like to survey state-of-the-art DL-based methodologies for fish habitat monitoring. This Chapter presents the second part of the literature review component of this thesis.

3.1 Introduction

Proper understanding of our planet and its ecosystems is not possible unless suitable tools are developed to explore and learn about our largest ecosystem, the marine environment. Computer Vision (CV) technology through deployment of its underwater cameras can help us better comprehend and manage remote marine fish habitats. However, due to the sheer volume of their visual data, manual processing is time- and cost-prohibitive, requiring a new radical shift in data analysis, through advanced technologies such as Deep Learning (DL).

DL is at the frontier of computer vision. Its deep neural network architectures are capable of learning complex mappings from high-dimensional data to interpretable feature representations, hence, DL has been successfully applied to various challenging computer vision tasks such as semantic image segmentation [177–181], visual object detection [77, 182–184], and tracking [185–188]. These applications have the potential to radically alter the way we interact with the world through computers. Recently, the applications of DL and its underlying DNNs for underwater visual processing have received significant attention [9, 24, 79, 147, 151, 161, 189–191].

The main advantage of deep learning is its ability to learn features in different data types, such as underwater fish images, through end-to-end training. Training of DNNs is often thought to be easy. Many frameworks take delight in providing few lines of code that solve some CV tasks, providing the misleading impression that all that is needed is then plug and play, using some general Application Programming Interfaces (APIs). In these APIs, the developers have lifted the burden from us and, in doing so, disguised the complexity behind a few lines of code needed to achieve the task at hand. The framework developers have achieved the purpose of ”providing a few lines of code” but we, the end-users, have been fooled into believing we need to spend only a few hours learning the intricacies of the provided APIs.

However, when it comes to training a DL algorithm, things become more complicated. The task of training a DNN is actually as complicated as the problem it is intended to

solve. In fish monitoring, for example, the number of input images you use, how you pre-process your images, how you build your models, how you fine-tune the model (using dropout or regularisation, for example), how you extract the features, how you combine them to produce final predictions, what metric you use to report your model performance, and your choice of which layer to extract features from to feed to your classifier, are among some of the many variables to consider when training a DNN. You can include any number of variations on these factors to further optimise your model and achieve the best possible accuracy.

Due to the above intricacies, most of the time DNNs are not simply an "off-the-shelf" technology that works with all kinds of datasets, even those similar to the one that has been meticulously customised for it. The fact that training a customised high-performance DNN is rigorous and challenging is now widely accepted. However, this challenging process can be facilitated by being patient, paying attention to details, and working systematically. Developing customised DNNs with a specific application, for example, for underwater fish monitoring, should follow the same systematic steps of developing any other computer vision applications (e.g. detection of vehicles in traffic). The only difference lies in the type of data being fed to the DNN.

In this paper, we first present a tutorial that covers the background of DL to help understand the above-mentioned common DL terminologies. The tutorial also provides a comprehensive overview of the essential systematic steps to help better develop a supervised DL model, with a focus on underwater fish habitat monitoring.

In the second part of the paper, we survey state-of-the-art research and development on the use of DL for fish monitoring. We synthesise the literature into four main categories covering the common CV tasks of classification, counting, localisation, and segmentation of fish images. We investigate different deep learning architectures and their performance. We also survey publicly available underwater fish image datasets. Finally, we provide a comprehensive overview of the challenges in applying DL to marine fish monitoring domains. We also draw a roadmap for future research works.

Although a number of previous relevant review articles [1, 58, 59, 192–195] exist, our paper has a different approach and motivation that compliments prior surveys. Compared to [58], which provides a survey of the general domain of ecological data analysis, covering a wide array of studies on plankton, fish, marine mammals, pollution, and nutrient cycling, we focus only on fish monitoring. We also provide a detailed analysis of fish datasets and comprehensively review the literature on four key tasks in underwater fish video and image processing. This detailed analysis and review are not provided in [58],

or any of the previous works, making our paper useful for readers who would like to study fish monitoring using DL in more detail and depth, while seeing a comprehensive literature review.

In addition, [59] provides a review of studies on fish condition, growth, and behaviour monitoring in aquaculture settings. It briefly covers and reviews various DL architectures and their aquaculture applications, unlike the present communication that is focused mainly on CNN and provides a detailed survey and analysis of the underwater fish monitoring literature.

The work presented in [192] covers the general domain of Machine Learning, as opposed to the specific domain of DL in our paper. This is done for aquaculture applications as wide as fish biomass and behaviour analysis to water quality predictions, while also briefly covering and reviewing fish classification and detection methods.

A survey of computer vision models for fish detection and behaviour analysis in digital aquaculture is provided in [193]. An interested reader should study [193] before reading our paper, due to the background technical details provided on image acquisition, which are key to developing effective DL datasets and models, as we discussed in our paper.

Furthermore, the DL-based studies presented in [194] and [195] are mainly around the two specific tasks of underwater fish tracking, and underwater object detection, respectively. These applications are different to our study. However, since our underwater fish monitoring task are related to these applications, our paper can complement these works.

In [1], we have provided a historical survey of fish classification methods between the years 2003-2021. These methods cover traditional CV techniques and modern DL methods, only for fish classification in underwater habitats and not for the general domain of underwater fish habitat monitoring.

This paper covers the use of deep learning in underwater fish monitoring. Section 3.2 covers the basics of deep learning, including neural networks, convolutional neural networks, and supervised learning. Section 3.3 provides an overview of the development process of deep learning models, from training to deployment. Section 3.4 discusses the applications of deep learning in underwater fish monitoring, including classification, counting, localization, and segmentation. Section 3.5 discusses the advantages and disadvantages of the application of DL to fish habit monitoring. Section 3.6 explores the challenges of underwater fish monitoring, such as environmental factors, model generalization, and limitations of available datasets. Section 3.7 presents potential opportunities for deep learning in underwater fish monitoring, including knowledge distillation, merging image data from multiple sources, automatic fish phenotyping, and visual monitoring

of fish behaviour and movements. Finally, Section 3.8 summarizes the study's main findings and provides concluding remarks.

3.2 Deep Learning

This section discusses the basics of deep learning [1], a sub-field of machine learning, and its utilization of multi-layered neural networks to automatically learn input features. It also introduces Convolutional Neural Networks (CNNs) and their efficient learning of deep features for image processing, making them suitable for underwater fish monitoring [1].

3.2.1 Neural Networks

Neural networks are a type of computational model that are inspired by the structure and function of biological neural systems in animals. They consist of basic processing units called neurons that take input signals, apply a function to them, and produce an output. In a neural network, the neurons are organized into layers, with each layer performing a specific type of computation. The layers are typically arranged in a hierarchical fashion, with the input layer receiving raw data and the output layer producing the final result.

The activation function of a neuron is the mathematical function that determines whether or not the neuron "fires" or produces an output signal based on the input signals it receives. One common activation function is the sigmoid function, which is a non-linear function that maps the input to a value between 0 and 1. This function is useful for classification tasks, such as image classification, where the output of the neuron can be interpreted as a probability.

Bias nodes are another important component in neural networks. These nodes are like neurons, but they do not receive input signals. Instead, they have a fixed input value of 1 and a weight associated with them. The bias value is added to the sum of the input-weight products to increase the flexibility of the model. In other words, bias nodes allow the neural network to adjust the output even when all input features are equal to zero.

Different types of loss functions are used for different types of tasks. For classification tasks, such as image classification, the cross-entropy loss is a common choice. This loss function measures the difference between the predicted probability of the correct class and the actual probability. Hinge loss is another type of loss function that is commonly used for classification tasks, where the correct class score should be higher than the sum

of the scores for all other classes by some margin.

Regularisation is a technique used in neural network learning to prevent overfitting by discouraging complex mapping functions or models. This technique involves adding a regularisation term to the general model loss function, which takes into account the loss function value for all the training dataset examples. The two most common forms of regularisation are L1 and L2, with L2 being the sum of the square of the weights, and L1 being the sum of the weights.

Optimisation

In supervised learning, the learning task can be reduced to an optimisation problem in the form of

$$\theta^* = \arg\min_{\theta} g(\theta), \tag{3.1}$$

where θ is a parameter vector, at which the loss function $g(\theta)$ that usually represents the average loss for all training examples, reaches its minimum. g can be represented as

$$g(\theta) = \frac{1}{n} \sum_{i=1}^{n} L\left(f_\theta\left(x_i\right), y_i\right), \tag{3.2}$$

where (x_i, y_i) represents a (input, desired output) training pair.

Similarly, in DL, an optimisation method is used to train the neural network by minimising the error function E that is defined as

$$E(W, b) = \sum_{i=1}^{m} L\left(\hat{y}_i, y_i\right) \tag{3.3}$$

where W and b are the weights and biases of the network, respectively. The value of the error function E is thus the sum of the mean squared loss L between the predicted value $\hat{y}$ and true value y, for m training examples. The value of $\hat{y}$ is obtained during the forward propagation step and makes use of the previously-mentioned weights and biases of the network, which can be initialised in different ways. Optimisation minimizes the value of the error function E by updating the values of the trainable parameters W and b.

The error function E is usually minimised by using its gradient slopes for the parameters. The most commonly used optimisation method is *Gradient Descent* [196], in which the gradient is optimised by calculating a matrix of partial derivatives (computed using backpropagation, as detailed in the next subsection). These derivatives provide the slope of g simultaneously at each dimension of θ. Therefore, the gradient-based optimizer is

used to iteratively update the network weights in the direction of the steepest descent of the loss function, with the aim of reducing the training loss to as low a value as possible. This is achieved by subtracting a small quantity from each weight in the direction of the negative gradient of the loss function. While the ultimate goal is to find a good local minimum of the loss function, the non-convexity of the loss function makes it difficult to search for the global optimum directly. Instead, the optimizer seeks to improve the network's performance on the training data, while also ensuring that the validation loss remains low, which indicates that the network is generalizing well to new data.

Backpropagation

Backpropagation is probably the most important part of learning in neural networks. It is performed after a forward propagation or pass, in which a subset of the training dataset (named a batch) $\{(x_i, y_i)\}_{i=1}^{m}$ and the current network parameters θ are used to calculate the final layer output and the loss. During the forward pass, the data input is passed to the first layer to process according to its activation function and their values are passed on to the next layer, hence the term "forward pass". After the forward pass and calculating the final layer loss, backpropagation happens, through which we start to calculate the loss backwards, layer by layer, and the layer derivatives are then "chained" by the local gradients to minimise the overall loss, g.

Overall, neural networks are a powerful and flexible tool for a wide range of machine learning tasks, and their components, including neurons, activation functions, bias nodes, and loss functions, are essential to their success.

3.2.2 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a type of Deep Neural Network that are particularly powerful for computer vision tasks. They work by applying a convolution (filtering) operation on the input data through several convolution layers. This extracts useful features from the input data by sliding convolution filters across the input image represented to the network as matrices. One of the first and most successful examples of CNNs in computer vision was AlexNet proposed in 2012 [103] . Since then, many different variations of CNNs have been proposed, revolutionizing image processing in different domains.

A typical CNN architecture consists of convolutional layers, pooling layers, non-linear activation layers, and final output layers, as shown in Figure 3.1. The building blocks and

layers of a typical CNN include Convolutional Layers, Batch Normalisation, Activation Layer, Pooling Layer, Dropout, and Fully Connected Layers. Convolutional Layers apply a filtering operation on input matrix data to generate a feature map. Batch Normalisation is used to normalise the learning of the network across the current set of training data to improve the speed of learning and convergence of the deep learning model. Activation Layers increase the non-linearity of the convolutional layer output to learn complex data. Pooling Layers reduce the size of the feature map and improve the efficiency of computation. Dropout is used to avoid overfitting the training data. Fully Connected Layers contain a small number of neurons and are the second-last layer of a CNN, before the output layer.

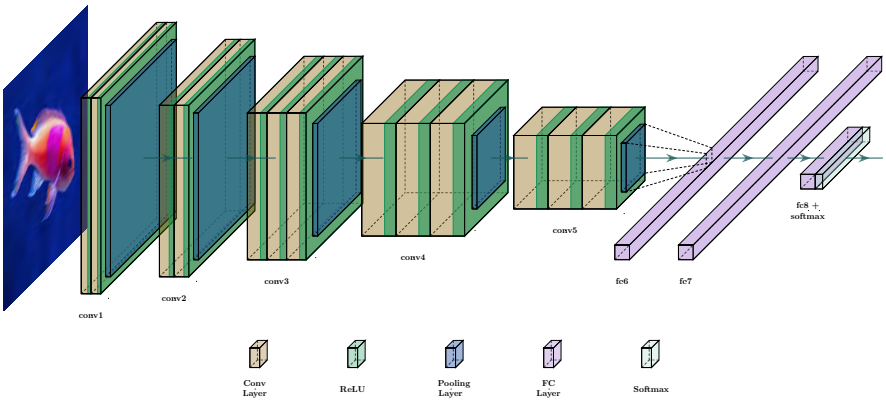

Figure 3.1: Schematic diagram of a CNN architecture used for the classification of fish images. The architecture consists of five convolutional layers that include the batch norm operation within them, followed by pooling layers (conv1-conv5). In this model, the feature maps from convolutional layers are pooled through pooling layers and then flattened through two fully connected layers (fc6 and fc7). The classification output is the result of a fully connected layer and a softmax activation layer (fc8+softmax).

3.2.3 Supervised Learning

There are two main approaches to learning in general DL. These include unsupervised and supervised learning. Unsupervised learning is often used to discover the structure and composition of the input and output domains without explicit and supervised target domain. This approach enables generalisation from one input domain to another by transforming data representations that are not directly related to the data distribution of target domain.

The supervised learning approach, on the other hand, is designed to explicitly map data from the input domain to its output domain via training pairs that exhibit matching representations. These pairs are carefully crafted by a human (supervisor), hence the name. The training process of supervised learning can suffer from instability and is less effective than the unsupervised learning method, because it learns with an accurate target distribution without domain-specific knowledge.

Supervised deep learning uses a subtle deep neural network mechanism to extract useful features from large amounts of input training data that are labelled to show their desired output domain. The learning is done by using the repetitive backpropagation process [197] explained earlier, to adjust the DL architecture parameters (such as weights and biases) while keeping fixed its hyperparameters (such as the shape, number, and size of convolutional, pooling, and fully connected layers). The goal is to optimize the function f, which maps the input domain X to the output domain Y. While the architecture of the network is typically fixed during training, the optimizer adjusts the internal parameters of the network to achieve the best possible mapping of the input training data to their desired output.

3.2.4 Deep learning and Fish Monitoring

One of the applications of DL is fish monitoring, which is the process of observing and measuring fish populations and their habitats. Fish monitoring is important for understanding the ecology and biodiversity of aquatic ecosystems, as well as for managing fisheries and aquaculture. DL can help with fish monitoring by providing accurate and efficient methods for fish classification, detection, counting, tracking, behaviour analysis, health assessment, and so on. DL can also handle complex underwater environments that pose challenges for traditional image processing techniques, such as low visibility, noise, distortion, illumination variation, etc.

The data for DL-based fish monitoring can come from various sources, such as underwater cameras, sonar, drones, satellites, etc. The data can be collected in different scenarios, such as shallow or deep water, fresh or marine water, natural or artificial habitats, etc. To improve the performance and robustness of DL-based fish monitoring systems, domain knowledge such as fish biology, ecology, and aquaculture management can be integrated and other technologies can be combined with DL algorithms [59]. These include hardware technologies: such as sensors, communication devices, storage devices, etc. Software technologies: such as data augmentation, feature extraction, model optimization, etc.

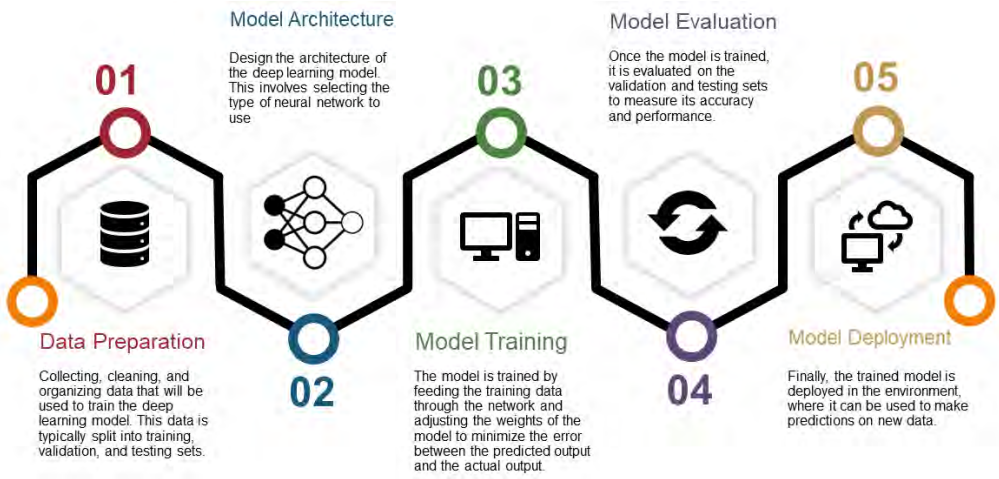

Figure 3.2: A schematic diagram showing the steps and components required for training a deep learning model.

The goals and effects of applying DL to fish monitoring are manifold [36]. Some examples of these goals are:

- Enhancing scientific understanding of aquatic ecosystems and their dynamics
- Improving fisheries management and conservation by providing reliable data on fish stocks and their distribution.
- Increasing aquaculture productivity and profitability by optimizing feeding strategies, reducing disease outbreaks, and preventing escapes and predation.
- Reducing human intervention and labour costs by automating fish monitoring tasks.
- Promoting public awareness and education on aquatic biodiversity and sustainability.

In summary, DL is a promising technique for fish monitoring that can provide automated solutions for various tasks related to fish identification, measurement, localization, and segmentation. By combining DL with other technologies such as sonar or drones, fish monitoring can be performed more effectively and efficiently in different underwater environments.

3.3 Developing Deep Learning Models

A comprehensive overview of the essential systematic steps for training a DL model is summarised in Figure 3.2. Even though these steps are general in DL training, we included useful tips arising from our experience in developing DL applications in various domains from medical imaging to marine science applications. Nevertheless, we put an emphasis on the development of DL for underwater fish habitat monitoring.

3.3.1 Training Dataset

The available training data is essential for developing an efficient DL model. Datasets are becoming increasingly crucial, even more so than algorithms. Perhaps, the most important factor when considering a supervised learning dataset is its size. The requirement for a large training dataset to achieve high accuracy is often a big obstacle. Because visual algorithms are trained by pairs of images and labels, in a supervised manner, they can only identify what has already been given to them. As a result, depending on the project, the number of objects to identify, and the required performance, training datasets might contain hundreds to millions of images. However, smaller training datasets with only a few hundred samples per class may also achieve good results [17, 42, 198, 199]. Nevertheless, the larger the training dataset, the greater the recognition accuracy.

Because of the scarcity of datasets and the difficulty of acquiring reliable data, approaches for boosting the accuracy rate from small samples will inevitably become a focus of future studies. The problem of limited sample data can be also alleviated by transfer learning [200–202]. Furthermore, data augmentation will become increasingly critical. Section 3.6.3 covers some challenges of limited data and some approaches to address these challenges.

The second factor to consider when preparing a dataset for DL training is having a balance. This is critical to ensure that each class to be identified contains a sufficient number of instances to minimise class imbalance biases. These biases happen when the DL favours one or more classes due to seeing them more often when being trained.

Also, the training dataset is typically divided into two subsets, the training subset for efficiently training the model and the validation/test subset for assessing the trained model's performance. For the training subset, a subset of the training dataset is reserved for training the model. If the training subset is too large, it can prolong the model training. If, on the other hand, the training subset is too small, the resulting model may not generalise well to unseen inputs. The validation/test subset is typically used to avoid overfitting,

which is a common problem in machine learning and happens when the developed model simply memorises the inputs rather than properly learning them. Cross-validation is another widely used methodology for testing a DL model's training performance, by splitting the training dataset into multiple mutually exclusive subsets of training and testing data. One method of cross-validation is called $k - fold$ cross-validation, in which the training dataset is split into k equally sized subsets. In this method, $k - 1$ folds are used for training the model, while the remaining fold is used to test the learning performance. This process is repeated until all the folds have been used once as a test/validation set.

In addition to the above, it is usually vital to, initially and before embarking on code development, perform a comprehensive inspection of the dataset. This will help to clean the dataset, for instance by finding and removing duplicate data instances. It also helps identify imbalances and biases, as well as data distribution, trends, or outliers, which will help in better model design and understanding of possible wrong DNN predictions.

Fortunately, in the domain of fish habitat monitoring, researchers currently have access to a variety of datasets. Table 3.1 lists publicly available underwater fish datasets, their sources, and where to get them, in addition to a summary of their features, their labels, and their sizes. The main point to note about these datasets is that they differ in both size and number of features. Although the number of these fish datasets is still small (17), the diversity of aquatic species they cover is already quite wide. They cover a large number of aquatic species, as indicated in Fig. 3.3. Moreover, each dataset features a different number of images that have varying resolutions. For each image, there is also a ground truth annotated by a human expert, which make them very useful. For instance, these datasets can be used by researchers to test their DL models or to pre-train them, as the first step, for their more specific fish monitoring tasks.

After preparing the training dataset or utilising alternative approaches to addressing insufficient data challenge, one can start developing their DL model using a machine-learning development framework.

3.3.2 Development framework

The rapid evolution of DL has led to the creation of a vast number of development libraries and packages that enable the setting up of DNNs with insignificant effort. Usability and availability of resources, architectural support, customisability, and hardware support are all various benefits of using existing machine-learning frameworks. The most commonly used frameworks are PyTorch, Tensorflow, MATLAB, Microsoft Cognitive Toolkit (CNTK) and Apache MXNET. In the context of DL for marine research, as will

Table 3.1: Summary of some publicly available datasets containing fish for training and testing deep learning models.

Dataset	Summary	Labels	Dataset size	Website
A - **Deepfish**	Videos from coastal habitats in north-eastern and western Australia	fish/no fish	40k classification labels, 3.2k images with point-level annotations, 310 segmentation masks	github.com/alzayats/DeepFish
B - **Croatian Fish Dataset**	12 species of fish found in Croatian waters	species names	794 classification labels	www.inf-cv.uni-jena.de/fine_grained_recognition.html#datasets
C - **Fish in seagrass habitats**	RUV taken in Australian seagrass habitat of 2 species	species	9k classification labels, bounding boxes and segmentation masks	github.com/globalwetlands/luderick-seagrass
D - **Fish4Knowledge**	Fish detection and tracking dataset, 17 videos at 10 min long, rate of 5 fps.	fish/no fish	3.5k bounding boxes	groups.inf.ed.ac.uk/f4k/index.html
E - **Fish-Pak**	Image dataset of 6 different fish species from 3 locations in Pakistan	species	1k classification labels	data.mendeley.com/datasets/n3ydw29sbz/3
F - **Labeled Fishes in the Wild**	Rockfish (Sebastes spp.) and other species (non-fish) near the seabed	fish/non-fish	1k bounding boxes (fish), 3k (non-fish)	swfscdata.nmfs.noaa.gov/labeled-fishes-in-the-wild/
G - **OzFish**	Large data set comprising of 507 species of fish.	species, fish/no fish	80k labeled cropped images, 45k bounding box annotations (fish/no fish)	github.com/open-AIMS/ozfish
H - **QUT Fish Dataset**	468 species in varying ex-situ and in-situ habitats.	species name	4k classification images	www.dropbox.com/s/e2xya1pzr2tm9xr/QUT_fish_data.zip?dl=0
I - **Whale Shark ID**	543 individual whale sharks (Rhincodon typus)	individuals	7.8k bounding boxes	http://lila.science/datasets/whale-shark-id
J - **Large Scale Fish Dataset**	9 different seafood types collected from a supermarket in Izmir, Turkey	species name	For each class, there are 1000 augmented images and their pair-wise augmented ground truths	www.kaggle.com/crowww/a-large-scale-fish-dataset
K - **NCFM**	Image dataset of 8 different fish species	species name	˜16000 classification images	www.kaggle.com/c/the-nature-conservancy-fisheries-monitoring/data
L - **Mugil liza sonar**	Sonar-based underwater videos of schools of migratory mullets (Mugil liza)	number of fish	500 counting images	zenodo.org/record/4751942#.YKzfUKgzayk
M - **MSRB Dataset**	Real underwater images without marine snow and synthesized with marine snow	NA	˜6000 images	github.com/ychtanaka/marine-snow
N - **WildFish**	1,000 fish categories	species name	˜54000 classification images	github.com/PeiqinZhuang/WildFish
O - **SUIM**	Image dataset of 8 different underwater objects	object name	˜1500 annotated images semantic segmentation mask	github.com/xahidbuffon/SUIM
P - **DZPeru fish-datasets**	Several species in varying ex-situ and in-situ habitats.	species name	˜17000 annotated images segmentation mask	github.com/DZPeru/fish-datasets
Q - **LifeCLEF**	10 different fish species	species name	˜1000 annotated videos	www.imageclef.org/

be shown later in Tables 3.3 to 3.5, PyTorch and TensorFlow are the dominant frameworks, while Matlab and Caffe have been used only in a few works. Overall, details such as the project needs and the programmer and developer preference should be taken into account, when choosing the development framework.

When the development framework is chosen, the next step is to find the most suitable network architecture for the task at hand. This sometimes depends on the framework, as some recent methods may not immediately be supported by all frameworks.

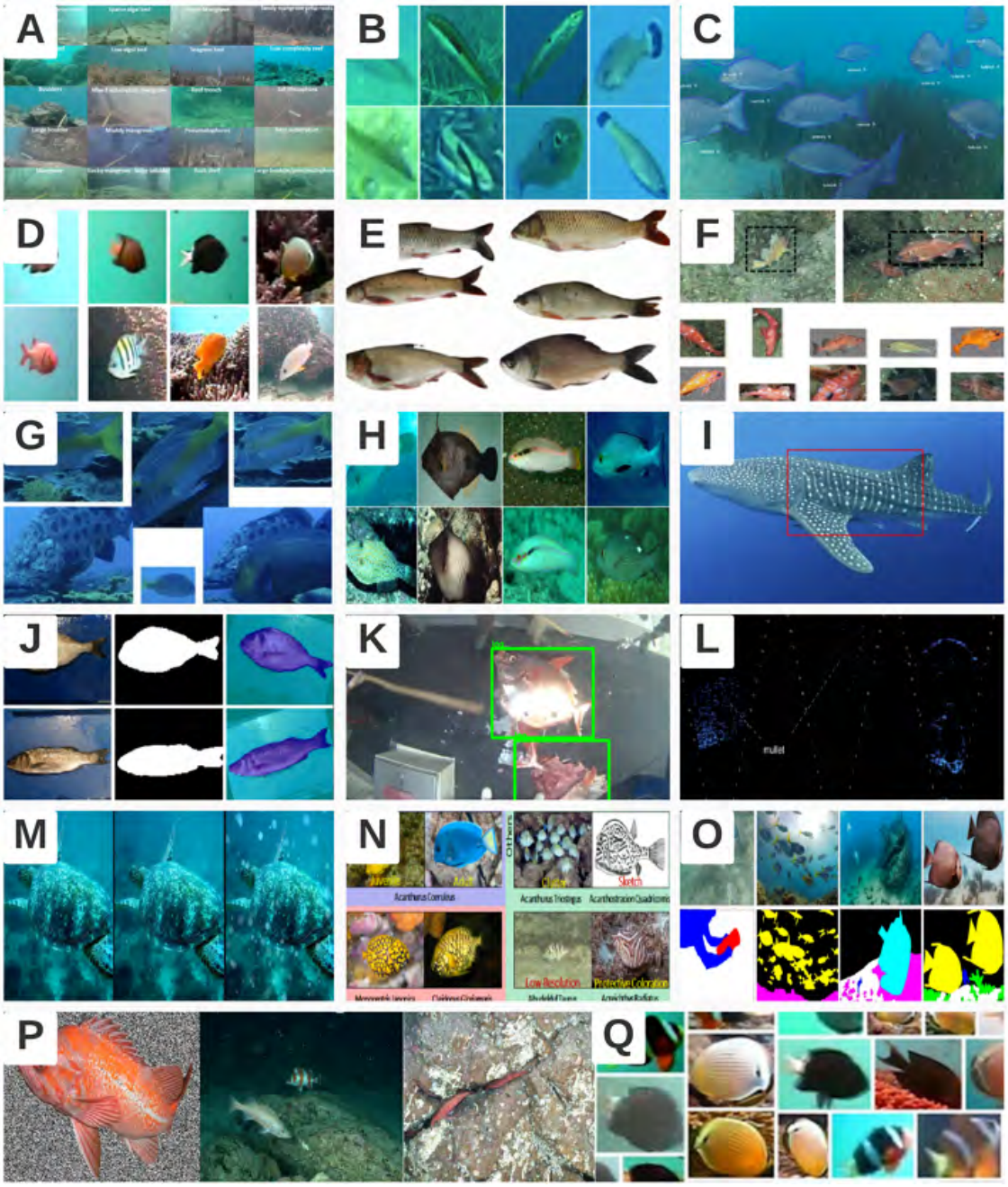

Figure 3.3: Sample images from publicly available datasets detailed in Table 3.1.

3.3.3 Network Architecture

Network architecture is the structure of the DL model, which depends on what it intends to achieve and its expected input and output. Therefore, the type of training dataset and the expected outcome influence the architecture's choice and its performance. DL network architectures can differ in a variety of ways such as the type and number of layers, their structure, and their order. Before selecting a network architecture, it is critical to understand the dataset you have and the task you are going to complete. For example, convolutional neural networks or CNNs are known to learn higher-order features, such as colours and shapes, from data within their convolution layers. Therefore, they are ideally adapted to image-based object recognition. On the other hand, Recurrent Neural Networks (RNNs) have the capability of processing temporal information or sequential data, such as the order of words in a sentence. This feature is ideal for tasks such as handwriting or speech recognition.

In the context of fish habitat monitoring, if you are working on a task that requires you to learn temporal information of the input sequence, for example fish image sequence analysis, the DL architecture you choose can be very important. For example, a CNNs architecture is more suited for *image-based* object recognition such as fish classification, while the RNN architecture is more suitable for tasks where the input sequence is temporal in nature such as generating fish habitat descriptions.

To find a suitable architecture, you first need to define your problem. This problem is defined by two questions: (1) What features will you extract? (2) How will you label these features? The features you extract are defined by your data. In other words, you are interested in the representation of the data you have. The number of features you choose to extract is defined by the task you are trying to solve. As described above, the DL architectures can learn features such as colours and shapes from image-based object recognition. Before trying to construct your network, you first need to decide what data type you will use and how will you encode the information. After you have defined your task, you should think about what features are important for the task. You will need to define this in order to construct your network. For example, if the features you want to extract are fish shape and fish location, then you could define a convolutional architecture. The features you choose to define should be a subset of all the features in the data. For example, for an image-based object recognition network, you would extract features such as fish species. However, your extracted features will also need to cover all the data. For example, you will also need features of the type of water or the type of background. It is important to take all these features into account when defining your network. For a

complete discussion on different DL architectures see [203].

3.3.4 Network Model

When a general network architecture is selected, the next step is to select, or sometimes develop, a network model of that architecture. For instance, when you decided to use a CNN, you can use different varieties of CNN models. The rule of thumb for selecting a CNN is to choose a model that results in a satisfactory training loss for your dataset. Creating an exotic and creative model is not recommended at this stage. It is usually recommended to avoid the temptation and choose a model big enough to overfit your dataset, and then regularise it properly to improve the validation loss.

For example, one may pick a well-known CNN model, e.g. ResNet, which can be used out-of-the-box, if their task is simple, e.g. fish classification. In later stages, they can customise their model to adequately capture their dataset. We show in Tables 3.3 to 3.5 in the next section that ResNet is the most commonly used model for fish counting (Table 3.3), fish localisation (Table 3.4), and fish segmentation (Table 3.5).

3.3.5 Training the model

After choosing the best model is time to set up a full train/validation pipeline. The below steps are recommended at this stage of development.

- Start with a simple model (i.e. a small number of convolutional layers) that can hardly go wrong and visualise the model performance metrics. Do not use an out-of-the-box large model like ResNet, just yet. It is recommended to plot training loss to see how the network is progressing during learning and if the loss is getting smaller. This also shows the speed of learning.

- To better understand the process, it is recommended to use a fixed random seed (for randomly initialising the network parameters) to ensure that the same results can be achieved when running the code twice.

- Do not perform any data augmentation at this stage as it may introduce errors. You can do data augmentation at a later stage after confirming that your network works properly. You can see a brief introduction to data augmentation and other methods at subsection 3.6.2.

- Use ADAM algorithm [204], which helps the learning by applying adaptive optimisation to the learning rate of the network.
- The learning rate is an important hyperparameter of a deep learning model. It is usually the most crucial value during training and should be configured using trial and error. Depending on the size of your dataset, a specific learning rate decay may be needed. The learning rate decay is a technique that allows the learning rate to fall during successive training epochs until it converges. A high learning rate at the start prevents the network from memorising noisy data, whereas decaying the learning rate improves complex pattern learning.
- Implement early stopping and monitor the learning process by looking at the training loss plot to prevent overfitting.
- Add complexity to your model gradually, e.g. add more layers or use off-the-shelf CNN models, and obtain a performance improvement over time.

3.3.6 Testing the model

When the model is trained, its accuracy and performance should be tested using the test subset of the training dataset. A test set can also be independent of the training dataset to evaluate the model performance. The main point to remember is that the test set should not have been used for the training or evaluation of the model, at all.

The model's performance should be measured by computing appropriate metrics suitable to the task at hand. A list of the most common metrics used in testing fish monitoring models is given in Tabel 3.2. For classification tasks, Classification Accuracy (CA), Precision and Recall rates are appropriate metrics, while F1-score, which is a combination of precision and recall, can provide a better measure of model performance and is used in fish counting and localisation tasks as shown in Tables 3.3 and 3.4. The Intersection-Over-Union (IoU) is the appropriate metric for segmentation tasks, while the mean average precision (mAP) metric suits pixel-wise localisation of fish in images. Looking at Tables 3.3 to 3.5, other metrics such as Mean Square Error (MSE) and Root MSE (RMSE) have also been used in the marine fish monitoring literature. These can be considered and used if required.

3.3.7 Fine Tuning the model

The performance and accuracy of the model could be improved if needed. The amount of this improvement is, though, strongly influenced by its current accuracy. This step may quickly become complicated, since increasing the model accuracy might require several steps such as adjusting the learning rate, collecting new data, or fully modifying the model's architecture. You should keep this fine-tuning step to a reasonable level. Otherwise, the model might overfit the data.

3.3.8 Deploying the model

Finally, the model deployment mode should be chosen. This depends on the application and the deployment requirements. The model can be deployed to run on a local or remote device (on a web server, a docker container, a virtual private server (VPS), etc). This will determine whether the results can be accessed remotely or only within the local network. It is recommended to use a cross-platform deployment method to avoid issues such as input/output data format, or the type of files used for storing data.

The most commonly used cross-platform model deployment method is Docker [205, 206], which is a virtualisation software that allows setting up and running other software environments on top of a base Linux distribution without the need to set-up virtual machines. Docker helps build, configure, and run applications using the same Docker file. Typically, Docker is the recommended approach for web applications. In this method, you can use Docker container or Docker host on your development machine. Docker container may be the easiest option for web applications. You can also deploy your network to a remote machine via Docker. The advantage of using a container is that you can share the development environment and run tests of your model using multiple docker containers. You can also install the Docker tool on your local machine to manage containers, so it is convenient.

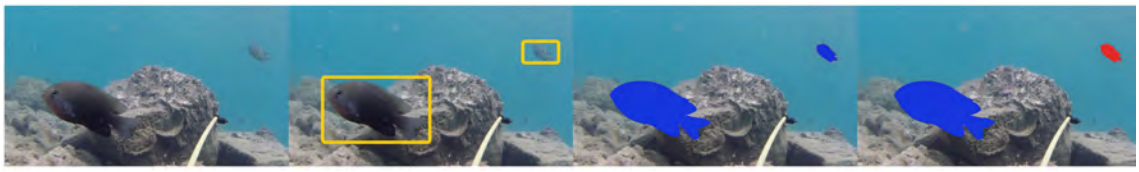

Figure 3.4: Illustration of four typical fish monitoring tasks. From left: Fish Classification (*i.e.* is there a fish in the image, or what type (class) of fish is in the image?); Fish Detection/Localisation/Counting; Fish Semantic Segmentation, and Fish Instance Segmentation.

Table 3.2: Performance metrics used to compare various surveyed works.

Performance Metric	Symbol Used	Description
Classification Accuracy	CA	The percentage of correct predictions. For multi-class classification, CA is averaged among all the classes. $CA = (TP + TN)/(TP + TN + FP + FN)$
Precision	P	The fraction of true positives (TP), to the sum of TP and false positives (FP). $P = TP/(TP + FP)$
Recall	R	The fraction of true positives (TP) to the sum of TP and false negatives (FN). $R = TP/(TP + FN)$
F1 score	F1	The harmonic mean of precision and recall. $F1 = 2 \times (P \times R)/(P + R)$
Mean Square Error	MSE	Mean of the square of the errors between predicted and observed values
Root Mean Square Error	RMSE	Is the square root of the mean of the square of all of the errors.
Mean Relative Error	MRE	The mean error between predicted and observed values, in percentage
L2 error	L2	Root of the squares of the sums of the differences between predicted counts and the actual counts
Intersection over Union	IoU	A metric that evaluates how similar the predicted bounding box is to the ground truth bounding box. by dividing the area of overlap between the predicted and the ground truth boxes, by the area of their union.
The maximum number	MaxN	MaxN, the maximum number of the target species in any one frame.
Mean average precision	mAP	Depending on the detection difficulty, the mean AP across all classes and/or total IoU thresholds are used.
Classification Error	CE	Is how often is the classifier incorrect and also known as "Misclassification Rate". $CE = (FP + FN)/(TP + TN + FP + FN)$

3.4 Applications of Deep Learning in Underwater Fish Monitoring

Deep learning has been widely used in marine environments with applications spanning from deep-sea mineral exploration [123] to automatic vessel detection [122]. However, we confine the scope of this paper to only marine fish image processing, which typically includes four tasks of classification, counting, localisation, and segmentation of underwater fish images, as shown in Fig. 3.4.

Here, the goal is to assist the reader in understanding the similarities and differences across these tasks and their relevant DL models and techniques. We provide a background of what each task involves, what previous works have been published toward addressing it using deep learning, and synthesise the literature on each task.

3.4.1 Classification

As its name infers, in visual processing, classification is the task of classifying images into different categories. There can be only two categories, i.e. a binary classification, in which the images are classified into two groups, e.g. ”fish” and ”no fish”, depending on the presence or absence of fish in an image (e.g. Deepfish dataset described in the first row of Table 3.1). The classification can also involve multiple ”classes” or groups. For instance, consider assigning different underwater fish images into different groups based on the species (e.g. FishPak dataset in Table 3.1) present in them.

Consider a manual procedure, in which images in a dataset are compared and relative ones are classified based on similar features, but without necessarily knowing what you are searching for in advance. This is a difficult assignment as there could be thousands of images in the dataset. Moreover, many image classification tasks involve images of different objects. It rapidly becomes clear that an automatic system, such as a DNN, is required to complete this task quickly and efficiently.

Classification is the most widely-used and -studied underwater image processing task using DL. In a previous work, we have covered the use of DNNs specifically for the task of underwater fish classification. We refer the reader to [1] for a comprehensive review of prior art on classification.

3.4.2 Counting

The purpose of the counting task is to predict the number of objects existing in an image or video. Object counting is a key part of the workflow in many major CV applications, such as traffic monitoring [207, 208]. In the context of marine applications and fish monitoring, counting may be used to map distinct species and monitor fish populations for effective conservation. With the use of commercially available underwater cameras, data gathering can be done more comprehensively. It is, however, difficult to correctly count fish in underwater habitats. To perform effective counting, models must understand the diversity of the items in terms of posture, shape, dimension, and features, which makes them complex. Meanwhile, manual counting is very time-consuming, costly, and prone to human error.

DL affords a faster, less expensive, and more accurate alternative to the manual data processing methods currently employed to monitor and analyse fish counts. Table 3.3 lists several of the recent DL techniques used for fish counting. Saleh et al [24] created a novel large-scale dataset of fish from 20 underwater habitats. They used Fully Convolutional Networks (FCNs) for several monitoring tasks including fish counting and reported a Mean Average Error (MAE) of 0.38%. DL has the potential to be a more accurate method for assessing fish abundance than humans, with results that are stable and transferable between survey locations. Ditria et al [10, 209, 210] compared the accuracy and speed of DL algorithms for estimating fish population in underwater pictures and video recordings to human counterparts in order to test their efficacy and usability. In single image test datasets, a DL method performed 7.1% better than human marine specialists and 13.4% better than citizen scientists. For video datasets, DL was better by 1.5% and 7.8% compared to marine and citizen scientists, respectively.

Despite this high potential, DL has not been thoroughly investigated for counting underwater fish. One possible reason for the lack of comprehensive research on fish counting is the scarcity of large publicly available underwater fish datasets. In addition, properly annotating fish datasets to train robust DL models is time-prohibitive and expensive. Although underwater fish counting is limited in the literature, several previous works have advanced the field in this area. For instance, Tarling et al [130] created a novel dataset of sonar video footage of mullet fish labelled manually with point annotations and developed a density-based DL model to count fish from sonar images. They counted fish by using a regression method [212] and achieved a MAE of 0.30%. Other researchers [131, 211] used sonar images as well because they present substantially different visual characteristics compared to natural images. Counting fish in sonar images, however, is substantially

Table 3.3: Summary of recent DL research works performing the task of fish counting

Article	DL Model	Framework	Data	Annotation/Pre-processing/Augmentation	Classes and Labels	Perf. Metric	Metric Value	Comparisons with other methods
A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis [24]	ResNet-50 CNN	Pytorch	Authors-created database containing 39,766 images for 20 habitats from remote coastal marine environments of tropical Australia and split to sub-dataset for four computer vision tasks: classification, counting, localization, and segmentation.	Each image was annotated by point-level and semantic segmentation labels	20 classes of 20 different fish habitat.	MAE	0.38	NA
Annotated Video Footage for Automated Identification and Counting of Fish in Unconstrained Seagrass Habitats [10]	ResNet-50 CNN	Pytorch	The dataset consists of 4,281 images and 9,429 annotations (9,304 luderick, 125 bream) at the standard high resolution (1920 x 1080 p).	Each image was annotated by drawing a bounding box and segmentation mask	2 classes of fish	F1	92%	NA
Automating the Analysis of Fish Abundance Using Object Detection Optimizing Animal Ecology With Deep Learning [209]	Mask R-CNN ResNet50	Pytorch	Authors-created database containing 6,080 fish images from 20 habitats from Tweed River Estuary in southeast Queensland	Each image was annotated by segmentation mask	1 class of fish	F1	Image (95.4%) Video (86.8%)	The computer's performance in determining abundance was 7.1% better than human marine experts and 13.4% better than citizen scientists in single image test datasets, and 1.5% and 7.8% higher in video datasets, respectively.
Deep learning for automated analysis of fish abundance: the benefits of training across multiple habitats [210]	Mask R-CNN ResNet50	Pytorch	Authors created five datasets, each consisting of 4700 annotated luderick, total of 23500 images	Each image was annotated by drawing a Polygonal segmentation masks around the region of interest (ROI)	1 fish class	F1	8792%	NA
Deep learning with self-supervision and uncertainty regularization to count fish in underwater images [130]	ResNet50 CNN	Tensorflow	Authors created a data set of 500 labelled sonar images from video sequences	Each image was annotated by dot annotation	3 classes of fish according to number of fish	MAE	0.30%	Comparison between DeepFish dataset 0.38% and authors' benchmark result and their model 0.30%.
Counting Fish and Dolphins in Sonar Images Using Deep Learning [131]	CNN	NA	Authors created a data set of 143 labeled sonar images from the Amazon River	Each image was annotated by counting number of fishes	35 classes for fish and 4 for dolphin	MSE	Fish 2.11% Dolphins 0.133%	Comparing four Network Architectures, DenseNet201, InceptionNetV2, Xception, and MobileNetV2
Counting Fish in Sonar Images [211]	CNN	NA	Authors created a dataset of 537 labelled sonar images from video sequences	Each image was annotated by dot annotation	1 class of fish	RMSE	16.48%	Comparison with other state-of-the-art approaches
Assessing fish abundance from underwater video using deep neural networks [189]	Faster R-CNN	Caffe	Authors created a dataset of 4909 labelled images from video sequences	Each image was annotated by drawing a bounding box	50 classes from 50 Different fish habitat.	mAP	82.4%	NA

different from counting fish in underwater video surveillance [189]. Unlike natural images, sonar images present unique visual characteristics and are in lower resolution due to the specific imaging forming principle.

Using DL, a computer can be taught to identify fish in underwater images, thus eliminating the subjectivity of humans in counting fish. However, its use for fish population and count analysis is dependent on the model performance on a set of well-defined performance metrics and parameters, which is in itself a challenge. In section 3.3, we discussed how one can train high-performance DL models, how the use of the current DL pipeline (and other methodologies) can be improved, and how future DL models can be designed for better assessing fish population including their abundance and their location, which is the subject of the next subsection.

3.4.3 Localisation

Object localisation is an essential task in CV, where the goal is to locate all instances of specified objects (e.g. fish, aquatic plants and coral reef) in images. Marine scientists assess the relative abundance of fish species in their environments regularly and track population variations. Various CV-based fish sample methods in underwater videos have been offered as an alternative to this tedious manual assessment. Though, there is no perfect method for automated fish localisation. This is mostly owing to the difficulties that underwater videos bring, such as illumination fluctuations, fish movements, vibrant backgrounds, shape deformations, and a variety of fish species.

To address these issues, several research works have been carried out, which are listed in Table 3.4. Saleh et al [24] have developed a fully convolutional neural network that performs localising of fish in realistic fish-habitat images with high accuracy. Jalal et al [128] introduced a hybrid method based on motion-based feature extraction that combines optical flow [213] and Gaussian mixture models [214] with the YOLO deep learning technique [215] to identify and categorise fish in unconstrained underwater videos using temporal information. They achieved fish detection F-scores of 95.47% and 91.2% on LifeCLEF 2015 benchmark [216] and their own dataset, respectively. Gaussian mixture is an unsupervised generative modelling approach that may be used to learn first and second-order statistical estimates of input data features [214]. Within an overall population, this is used to indicate Normally Distributed subpopulations. The weakness of Gaussian mixture is when trained on videos with some fish but no pure background, the fish are modelled as background as well, resulting in misdetections in subsequent video frames [217]. In order to compensate for the Gaussian mixture's weakness, optical flow

can be used to extract features that are solely caused by underwater video motion. The pattern of apparent motion of objects, surfaces, and edges in a visual scene generated by the relative motion of an observer and a scene is known as optic flow [213].

Knausgard et al [129] also implemented YOLO [215] for fish localisation. To overcome their small training samples, they employed transfer learning (explained in the next Section). The YOLO technique achieved Mean Average Precision (mAP) of 86.96% on the Fish4Knowledge dataset [218]. YOLO-based object detection systems have been also used in several other research to robustly localise and count fish [128, 129, 219]. To test how well Yolo could generalise to new datasets, [219] used it to localise fish in underwater video using three very different datasets. The model was trained using examples from only two of the datasets and then tested on examples from all three datasets. However, the resulting model could not recognise fish in the dataset that was not part of the training set.

Other CNN models have also been adapted to robustly detect fish under a variety of benthic background and illumination conditions. For instance, [183] and [220] used GoogLeNet [221], while [222] used an ensemble of Region-based Convolutional Neural Networks [223] that are linked in a cascade structure by Long Short-Term Memory networks [224]. In addition, Inception [225] and ResNet-50 [226] were examined in [227] for fish detection and recognition based on weakly-labelled images. Furthermore, [228] and [229] used Fast R-CNN (Region-based Convolutional Neural Network) [223] to detect and count fish.

Table 3.4 demonstrates that state-of-the-art methods (e.g. YOLO and Fast R-CNN) can achieve high accuracy in localisation tasks. These methods generally train object detectors from a wide variety of training images [230, 231] in a fully supervised manner. The drawback is that these models depend on instance-level annotations, e.g. tight bounding boxes need to be drawn around fish in training datasets. This is time-consuming and labour-intensive and makes the use of DL in marine research very challenging, if not impossible. In Section 3.6.3 we discuss how this critical issue can be addressed using weakly supervised localisation of objects, where only binary image-level labels showing the existence or absence of an object type are needed for training.

Similar to fish classification, counting, and localisation, fish segmentation, i.e. detecting the entire body of fish in an image is a critical task in marine research and applications. In the next subsection, we discuss how DL can be used to perform fish segmentation and how it is useful in marine research.

Table 3.4: Summary of recent DL research works performing the task of fish localization

Article	DL Model	Framework	Data	Annotation/Pre-processing/Augmentation	Classes and Labels	Perf. Metric	Metric Value	Comparisons with other methods
Marine Animal Detection and Recognition with Advanced Deep Learning Models [227]	ResNet-10 CNN	NA	The dataset is made of 73 videos from the public datasets Fish4Knowledge	Each image was annotated by drawing a bounding box	1 class of fish	F1	0.07%	NA
Fish detection and species classification in underwater environments using deep learning with temporal information [128]	Yolo - CNN	TensorFlow	The dataset is made of two datasets 93 videos from LifeCLEF 2015 fish dataset And an authors-created database containing 4418 videos	Each image was annotated by drawing a bounding box and species name	15 classes of 15 different fish species.	F1	LCF-15 95.47% UWA 91.2%	Comparison with other state-of-the-art approaches
Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system [217]	ResNet-152 CNN	TensorFlow	The dataset is made of 110 videos from two public datasets Fish4Knowledge and LifeCLEF 2015 fish dataset	Each image was annotated by drawing a bounding box	15 classes of 15 different fish species.	F1	87.44% and 80.02% respectively	NA
Temperate fish detection and classification: a deep learning based approach [129]	YoloV3 - CNN	Pytorch	total of 27230 images catalogued into 23 different species from the public datasets Fish4Knowledge	Each image was annotated by drawing a bounding box	23 classes of 23 different fish species.	mAP	86.96%	NA
Underwater Fish Detection Using Deep Learning for Water Power Applications [219]	YoloV3 - CNN	Keras - TensorFlow	Authors-created database of underwater video sequences for a total of 70000 train/test frame	Each image was annotated by drawing a bounding box	3 classes of fish	mAP	54.74%	NA
Coral Reef Fish Detection and Recognition in Underwater Videos by Supervised Machine Learning: Comparison Between Deep Learning and HOG+SVM Methods [183]	GoogLeNet CNN	NA	Authors-created database containing 13000 fish thumbnails from videos	Each image was annotated by drawing a bounding box	11 classes of 8 different fish species.	F1	98 %	Compare HOG+SVM With Deep Learning
Fish identification in underwater video with deep convolutional neural network [220]	GoogLeNet CNN	NA	20 videos from LifeCLEF 2015 fish dataset	Each image was annotated by drawing a bounding box	15 classes of 15 different fish species.	AP	81%	NA
Cascaded deep network systems with linked ensemble components for underwater fish detection in the wild [222]	RNN-LSTM	NA	Authors-created database containing 18 underwater video sequences for a total of 327 train/test frame	Each image was annotated by drawing a bounding box and species name	1 class of fish	F1	67.76%	Comparison with R-CNN Baseline
A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis [24]	ResNet-50 CNN	Pytorch	Authors-created database containing 39,766 images for 20 habitats from remote coastal marine environments of tropical Australia and split to sub-dataset for classification, counting, localization, and segmentation.	Each image was annotated by point-level and semantic segmentation labels	20 classes of 20 different fish habitat.	MAE	0.38	NA
Marine Organism Detection and Classification from Underwater Vision Based on the Deep CNN Method [228]	VGG16 -RCNN	NA	The dataset is obtained from the video provided by the Underwater Robot Picking Contest, test set contains 8800 images.	Each image was annotated by drawing a bounding box	3 classes of fish	mAP	91.2%	NA

3.4.4 Segmentation

Semantic segmentation task is to predict a label from a set of pre-defined object classes for each pixel in an image [232]. In the context of marine research, fish segmentation provides a visual representation of fish contour, which might be helpful for human expert visual verification or to estimate fish size and weight. Table 3.5 lists a number of research addressing the task of fish segmentation.

Saleh et al [24] developed a FCN model that performs fish Segmentation in realistic fish-habitat images with a high accuracy. Labao et al [233] proposed a DL model that can simultaneously localise fish, estimate bounding boxes around them and segment them using a unified multi-task CNN in underwater videos. Unlike previous approaches [234,235] that relied on motion information to identify fish body, their proposed method predicts fish object spatial coordinates and per-pixel segmentation using just video frames independent of motion information. Their suggested approach is more resilient to camera motions or jitters since it is not dependent on motion information, making it more suitable for processing underwater videos captured by Autonomous Underwater Vehicles (AUVs). Region Proposal Networks (RPN) [236] have been also used for fish segmentation in underwater videos [125]. RPN is a FCN that generates boxes around identified objects and gives them confidence scores of belonging to a specific class, simultaneously.

Computational efficiency is essential in the autonomy pipeline of visually-guided underwater robots. For this reason, [28] developed SUIM-Net, a fully-convolutional encoder-decoder model that balances the trade-off between performance and computational efficiency. On the other hand, for higher performance, [126] proposed Dual Pooling-aggregated Attention Network (DPANet) to adaptively capture long-range dependencies through a computationally friendly manner to enhance feature representation and improve not only the segmentation performance, but also its computational resources and time.

All previously discussed models use fully-supervised methods that require a large amount of pixel-wise annotations, which is very time-consuming and expensive, because a human expert must segment and label, for example, each fish in an image. To overcome this serious issue, weakly-supervised semantic segmentation models are used. These models do not need to be trained with pixel-wise annotation [238]. However, due to a lower level of supervision, training weakly-supervised semantic segmentation models is often a more challenging task. Applying weakly labelled ground truth derived from motion-based adaptive Mixture of Gaussians Background Subtraction, [237] managed to get an average precision of 65.91%, and an average recall of 83.99%. Recently, several other weakly-supervised methods have been introduced to overcome the cost of a large

Table 3.5: Summary of recent DL research works performing the task of fish segmentation

Article	DL Model	Framework	Data	Annotation/Pre-processing/Augmentation	Classes and Labels	Perf. Metric	Metric Value	Comparisons with other methods
A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis [24]	ResNet-50 CNN	Pytorch	Authors-created database containing 39,766 images from 20 habitats from remote coastal marine environments of tropical Australia and split to sub-dataset for classification, counting and localization, and segmentation.	Each image was annotated by point-level and semantic segmentation labels	20 classes of 20 Different fish habitat.	mIoU	0.93%	NA
Weakly supervised underwater fish segmentation using affinity LCFCN [9]	ResNet-CNN	Pytorch	Public DeepFish dataset [17]	Each image was annotated by segmentation labels	20 classes of 20 Different fish habitat	mIoU	0.749%	NA
Simultaneous Localization and Segmentation of Fish Objects Using Multi-task CNN and Dense CRF [233]	ResNet-CNN	TensorFlow	Authors-created dataset containing 1525 images from ten 10 different sites in central Philippines	Each image was annotated by drawing a bounding box and segmentation labels	1 class of fish	AP	93.77%	NA
Semantic Segmentation of Underwater Imagery: Dataset and Benchmark [28]	VGG16 -CNN	Keras - TensorFlow	Authors-created dataset containing 1525 images of 8 object categories	Each image was annotated by segmentation labels	8 classes of 8 different object categories.	mIoU	84.14%	NA
DPANet: Dual Pooling-aggregated Attention Network for fish segmentation [126]	ResNet-50 CNN	Pytorch,	Two public datasets DeepFish [17] and SUIM [28]	Each image was annotated by segmentation labels	20 classes: 20 Different fish habitat.	mIoU	91.08%, 85.39%	Comparison with other state-of-the-art approaches
Weakly-Labelled semantic segmentation of fish objects in underwater videos using a deep residual network [237]	ResNet-FCN	TensorFlow	Authors-created dataset containing several underwater videos from six different sites in Verde Island Passage, Philippines.	Each image was annotated with weakly-labelled ground truth derived from a motion-based background subtraction (BGS)	1 class of fish	AP	65.91%	NA
Improved deep learning framework for fish segmentation in underwater videos [125]	ResNet-CNN	TensorFlow	Two datasets extracted from the Fish4Knowledge to produce 2000 frames	Each image was annotated by drawing a bounding box and segmentation labels	15 classes of 15 different fish species.	AP	95.20%	NA

amount of pixel-wise annotations. These new methods include bounding boxes [239,240], scribbles [241], points [9, 242], and even image-level annotation [178, 182, 243–245]. Since weakly-supervised methods are integral to the success of important DL-based segmentation tasks, in Section 3.6.3, we discuss them further.

In the previous subsections, we discussed how DL is useful in a number of key applications in fish habitat monitoring. In the following Section, we discuss the many challenges on the way of developing DL models for such applications.

3.4.5 Acoustic and Sonar Data

Acoustic and sonar data are valuable sources of information for monitoring fish habitats and behaviours. Acoustic methods use sound waves to detect, identify, and quantify fish in various aquatic environments [246]. Sonar systems emit sound pulses and receive echoes from objects in the water, such as fish. By analyzing the characteristics of the echoes, such as frequency, intensity, and shape, sonar systems can provide information about fish size, shape, orientation, density, and movement.

Acoustic methods have several advantages over other techniques for fish monitoring, such as visual observation or net sampling [247]. Acoustic methods can cover large areas and depths quickly and efficiently; they can operate in turbid or dark waters where visual methods are ineffective; they can provide continuous data over long periods of time; they can minimize disturbance to fish and their habitats; and they can be integrated with other sensors or platforms for multidisciplinary studies.

Acoustic and sonar data can be combined with other technologies such as GPS and environmental sensors to provide a more complete picture of fish behaviour and their habitat. For example, the combination of acoustic and sonar data with GPS allows researchers to track fish movements and habitat use, while the integration of environmental sensors can provide information on water temperature, salinity, and other important environmental factors that may influence fish behaviour.

One of the challenges of acoustic methods is to accurately classify fish species based on their acoustic signatures [248]. Different species may have similar acoustic characteristics due to their morphology or behaviour. Moreover, environmental factors such as noise, reverberation, or multipath effects may degrade the quality of the acoustic data. Therefore, advanced signal processing and machine learning techniques are needed to improve the performance of acoustic classification. In addition, the deployment of acoustic and sonar sensors in natural environments can be challenging and expensive, which may limit the availability of data for the training and validation of DL models [249].

Acoustic and sonar data combined with DL techniques offer a powerful tool for monitoring fish habitats and behaviours in a non-invasive and efficient way [250]. By using this tool, fisheries scientists and managers can gain insights into fish ecology, distribution, abundance, migration patterns etc., which can help them make informed decisions for sustainable fisheries management.

3.4.6 Automatic Fish Phenotyping From Underwater Images

Automatic fish phenotyping, i.e. extracting their weight, size, and length, in their natural habitats can provide invaluable information in better understanding marine ecosystems and fish ecology [58]. Although many studies have addressed fish monitoring in aquaculture and fish farm settings [59, 192], monitoring fish for measurement in natural habitats remain mostly unexplored, and can be investigated in future research. This research should address problems such as low visibility and light, fish occlusion and overlap, which are shared with aquaculture monitoring. However, other problems unique to natural habitats such as cluttered background environments and underwater distance measurement should be addressed too. One study addresses fish species identification in an underwater video for marine monitoring applications, using a hierarchical CNN model that incorporates targeted data augmentation techniques [251]. Automated imaging has also been used to obtain phenotypic data on growth and body colour [252].

3.4.7 Visual Monitoring of Fish Behavior and Movements

Although some telemetry and satellite tracking devices can be used in limited settings [253], fish monitoring in their natural habitats over a period of time is not achievable using these techniques mainly due to the hostile underwater signal communication medium [174]. For instance for tracking fish movements, schooling, and behaviour, new visual monitoring techniques should be devised. A possible direction for future studies is to devise a better understanding of fish vision characteristics [136] and their implications in the current and next generation of automated DL-based tracking systems [194] and marine object detection [195]. An example of an alternative tracking method is presented in [254], where the image-based identification and tracking method for fish is designed based on biological water quality monitoring. To improve the fish tracking task, some techniques can also be combined with visual image enhancement algorithms. For instance, when the image enhancement methods are used, the underwater images can be corrected for distortion and noise, and the fish tracking task can be easily performed.

In [255], the authors studied the potential of underwater fish monitoring by using visual and underwater sensing methods.

Another challenging research area is developing novel underwater fish tracking algorithms, using DL or other technologies, with low power consumption and real-time speed. For this, various hardware technologies and techniques used in other domains such as biomedical applications [47] can be explored. Of course, any automated vision-based tracking system should be validated through real-world trials, which is a significant undertaking requiring many resources, in order to ensure the accurate and real-time tracking of fish.

There have been several recent studies on the visual monitoring of fish behaviour and movements. For example, some studies surveyed the application of computer vision technology in analyzing fish behaviour and fish monitoring [256–258]. Another study demonstrated an integrated object detection and tracking pipeline as a noninvasive and reliable approach to studying fish behaviour by tracking their movement under field conditions [26]. Another study explores how fish behaviour can be used as a proxy to measure the physiological states of fish under different environmental stressors, such as pollutants, temperature changes, and social interactions [259].

3.5 Advantages and Disadvantages of the Application of DL to Fish Habit Monitoring

Deep learning has been applied to various fields, including fish habitat monitoring [1]. The application of DL in fish habitat monitoring has several advantages that make it an attractive option for researchers and practitioners. One of the main advantages is its ability to handle complex data. DL models can learn complex patterns and relationships in the data, making them ideal for analyzing large datasets with numerous variables [210]. This ability is particularly useful in fish habitat monitoring, where numerous variables such as water temperature, dissolved oxygen, and water quality can influence the fish's behaviour and habitat.

Another advantage of using DL in fish habitat monitoring is the potential to automate the monitoring process. Traditional fish monitoring methods involve manual data collection and analysis, which can be time-consuming, labour-intensive, and expensive. With DL, data can be automatically collected and analyzed in real time, allowing for faster and more efficient monitoring. This automation can also reduce the likelihood of human error,

leading to more accurate and reliable results.

However, the application of DL to fish habitat monitoring also has some disadvantages. One of the main disadvantages is the need for large amounts of high-quality data to train the DL models effectively. The quality of the data can have a significant impact on the performance of the model, and the lack of high-quality data can lead to inaccurate results.

Another disadvantage is the complexity of the DL models themselves. DL models are often complex and difficult to interpret, making it challenging to understand how the model arrived at its conclusions. This lack of transparency can make it difficult for researchers and practitioners to verify the accuracy of the model's results.

In addition, DL models require significant computing power and storage, which can be expensive and require specialized infrastructure. This requirement can be a barrier for some researchers and practitioners who do not have access to the necessary resources.

Overall, while the application of DL to fish habitat monitoring has several advantages, it also has some drawbacks that need to be considered. To maximize the benefits of DL in fish habitat monitoring, it is crucial to address these challenges and develop strategies to overcome them.

3.6 Challenges in Underwater Fish Monitoring

Underwater fish monitoring presents a series of challenges for DL, which have been the focus of many research works [53]. In this section, we first introduce the major environmental challenges faced when developing underwater fish monitoring models. We then show that one of the approaches to properly address these environmental challenges is to use DL. However, DL training for fish monitoring has its own challenges, which will be discussed in detail.

3.6.1 Environmental challenges

In order to work in underwater environments, monitoring models must be able to recognise objects and scenes in complex, non-trivial backgrounds. This presents both a challenge in the development and training of these models and in robustly testing them. The main environmental challenges in underwater visual fish monitoring can be categorised as follows:

1. The environment is noisy including very large lighting variation. An object viewed from a distance is much less bright than a close-up object. These problems become

more acute when the background is not uniform.

2. Underwater scenes are highly dynamic, i.e. the scene's content and objects change very quickly. The background can change from being completely occluded to being visible and vice versa.

3. Depth and distance perception can be incorrect due to refraction. This is more severe for short distances.

4. Images are affected by water turbidity, light scattering, shading, and multiple scattering.

5. The image data are frequently under-sampled due to low-resolution cameras and power constraints underwater.

One of the main approaches used in literature to address these challenges is for the monitoring models to use hand-crafted features [79, 84, 87, 88, 92–96]. Hand-crafted features are defined by a human to describe a fish image. For example, a low-level feature can be the histogram of a texture or a Gabor filter response. As a more complex and representative feature, a mid-level feature can be a Scale-Invariant Feature Transform (SIFT) [90], or a Histogram of Oriented Gradient (HOG) [91]. However, human-defined features cannot be applied to other datasets, and the definition of a human-defined feature is a time-consuming task, which restricts real-time detection and requires manual effort. Moreover, hand-crafted features are limited by human experiences, which may contain noise and are difficult to design. For example, a SIFT descriptor doesn't work well with lighting changes and blur.

Therefore, a fish image is transformed into a feature space that a computer can understand. The feature space is often based on a combination of low-level image features (for example, colour distribution and gradient), and other features in the image such as edges, shapes, and textures. Models using hand-crafted features, however, do not perform well under varying environmental conditions, and the feature space cannot be easily or robustly created. Additionally, the features created are too low-level and cannot be easily used for processing images from different sources.

An alternative way to build prediction models capable of working in the presence of these significant environmental challenges is to use DNNs. However, training effective DNNs require resolving some other challenges, which we discuss in the below subsections. We also describe some of the approaches in literature addressing them. The re-

viewed approaches in addressing these common challenges can provide a quick reference for future researchers developing DL-based fish monitoring models.

3.6.2 Model Generalisation

Improving the generalisation abilities of DNNs is one of the most difficult tasks in DL. Generalisation refers to the gap between a model's performance on previously observed data (i.e training data) and data it has never seen before (i.e testing data). This is a fundamental problem, with implications for any applications using deep neural networks to process image data, videos, etc. This challenge is even more pronounced when more difficult tasks such as fish recognition in underwater environments.

Generalisation problem happens usually because during training the network over-fits to the training data. In other words, the weights of the network are adapted to produce a response that is best suited for reproducing the training examples. During testing, the network produces a response that is a compromise between the different training examples. This mismatch is a common cause of poor performance on test data, which is often referred to as a network over-fitting to the training data, even when the network has been trained for many epochs. The reason it occurs is that the network "memorises" the training data during the training. The training data can become quite large, consisting of hundreds of thousands or millions of examples. This makes the issue of network over-fitting quite significant. In the last few years, there have been significant research efforts toward solving the problem of over-fitting to improve model generalisation.

Previous works have shown that it is possible to prevent the network from over-fitting using techniques called regularisation [155]. There are also some theoretical techniques to make the network more robust to training data. Below, we provide a brief overview of some of these techniques and how they have been applied to solve the problem of deep network over-fitting to training data, to improve generalisation in DL.

- *Regularisation Term*: It is hypothesised that neural networks with fewer weight matrices can result in simpler models with the same capability as the complete model. A regularisation term is, therefore, added to the model loss function to remove some of the weight matrices components. The most popular methods of regularisation are L1 and L2. For example, [130] showed that incorporating uncertainty regularisation improves performance of their multi-task network with ResNet-50 [226] backend to count fish in underwater images.

- *Batch normalisation*: Introduced in Section 3.2.2 as part of the convolutional layer

in CNNs, batch normalisation was first introduced by [260] to decrease the effect of internal covariate shift. Internal covariate shift is the shift in the mean and covariance of inputs and network parameters across a batch of examples. Internal covariate shift can impede the training of deep neural networks. Batch normalisation is used in almost any DL model training, to improve the model generalisation. In the fish monitoring domain, for instance, [28] proposed an optional residual skip block consisting of three convolutional layers with batch normalisation and ReLU non-linearity after each convolutional layer to perform effective semantic segmentation of underwater imagery.

- *Dropout*: Introduced in Section 3.2.2 as a common operation in CNNs, dropout reduces the network dependency on a small selection of neurons and encourages more useful and robust properties and features of the dataset to be learnt. When working with a complex neural network structure, dropout is frequently recommended to introduce additional randomisation, which helps with the generalisation capability of the network. For example, [150] claimed that the inclusion of dropout layer has enhanced the overall performance of their proposed model for automatic fish classification.

3.6.3 Dataset Limitation

Preparing training datasets is one of the central and most time-consuming bottlenecks in developing DL models, which require a large amount of data, e.g. a variety of underwater fish images in different environmental conditions, which should also be labelled and analysed by humans for supervised learning. Due to these requirements, making a large dataset is most of the time, very challenging, which makes the datasets limited and small. However, when compared with DL models trained with a large dataset, the convergence speed and training accuracy of the models trained with small datasets are much lower. Generally, increasing the size of training datasets by adding more data to them is the classic way to accelerate the training and improve the accuracy of DL models, but it is expensive. Therefore, in recent years, researchers have tackled the dataset limitation challenge by devising new ways described below.

Data Augmentation

Data augmentation is a technique to increase the number of labelled examples required for DL training. It artificially enlarges the original training dataset by introducing various

transformations such as translation, rotation, scaling, and even noise, to the original data instances, to make new instances. It is particularly relevant to the challenge posed when the quantity or quality of labelled data is insufficient to train a DL model. At the same time, data augmentation can be used to reduce the probability of overfitting and increase model generalisability. In contrast to the techniques listed above for improving model generalisation, data Augmentation addresses overfitting from the source of the problem (*i.e.* the original dataset). This is done under the notion that augmentations can extract additional information from the original dataset by artificially increasing the size of the training dataset. It is also critical to consider data augmentation's "safety" (*i.e.* the possibility of misleading the network post-transformation). For example, rotation and horizontal flipping are typically safe data augmentation techniques for fish classification tasks [24, 67] but not safe on digit classification tasks, due to the similarities between 6 and 9. A data augmentation technique is to use the super-resolution reconstruction method [261] based on Generative Adversarial Network (GAN) [262] to enlarge the dataset with high-quality images. This has been previously used to improve small-scale fine-grained fish classification [157], and to increase the model's predictive performance (i.e. ability to generalise to new data) [42] for underwater fish detection and automatic fish classification [140].

Using augmentation techniques such as cropping, flipping, colour changes, and random erasing together can result in enormously inflated dataset sizes. For example, [28] used rotation, width shift, height shift, shear, zoom and horizontal flip for semantic segmentation of underwater imagery to significantly increase their dataset size. Another data augmentation technique used during training DL models are scale jittering, which has been used in [189] for assessing fish abundance in underwater videos. Gaussian filtering to blur images and different degrees of rotation for fish recognition in underwater-drone with a panoramic camera is another augmentation technique used in the marine monitoring domain [152].

However, augmentation is not always favourable, as it might lead to large overfitting in cases with very few data samples. As a result, it is critical to determine the best subset of augmentation techniques to train your DL model using a limited dataset.

Transfer Learning

Transfer Learning is preserving information obtained while solving one problem, and transferring the learned knowledge to another similar problem. For instance, one may initially train a network on a large object dataset, such as ImageNet that includes 1000 different object classes, and then utilise the learned network parameters from that train-

ing as the initial learning parameters in a new classification task, e.g. fish classification. In most cases, just the weights in convolutional layers are transferred, rather than the complete network, including fully connected layers. This is extremely useful since many image datasets have low-level spatial features and properties that are better learnt in massive datasets. For example, [156] presented unsupervised knowledge transfer to use their limited amount of training data in order to avoid time-consuming annotation for object detection in marine environmental monitoring and exploration.

Hybrid Features

DL architectures have demonstrated excellent capabilities in capturing semantic knowledge that is latent in image features. Handcrafted features, on the other hand, can provide specific physical descriptions if they are carefully chosen. In addition, attributes of natural images have been demonstrated to be described differently by CNN features and handcrafted features. This means a feature's discriminative ability may behave differently on different datasets. Therefore, these two types of features may complement each other for better learning.

However, increasing feature dimensions by fusing hand-crafted and DL-generated features can result in increased computational requirements. One way to avoid this is to initially utilise DL features for a particular dataset, and later add hybrid features to enhance the performance. As a result, when working with difficult datasets, such as uncommon and rare marine species, more sophisticated algorithms and techniques based on hybrid features may be required. In fact, several research groups have used such strategies to improve the performance of marine species recognition tasks.

For instance, [158] used texture- and colour-based hand-crafted features extracted from their CNN training data to complement generic CNN-extracted features and achieved a classification accuracy higher than when using only generic CNN features when classifying corals. A combination of CNN and hand-designed features have also been used in [159] for marine animal classification, again showing that their method achieves higher accuracy than applying CNN alone. In another work, [160] showed that aggregation of multiple features outperforms models using single feature-extraction techniques, for automated coral annotation in natural scenes .

Weakly-Supervised Learning

DL methods [109] have consistently achieved state-of-the-art results in a variety of applications, specifically in fully supervised learning tasks like classification and regression [263, 264]. Fully supervised learning methods create predictive algorithms by learning from a vast amount of training patterns, where each pattern has a label showing its ground-truth output [111]. Although the current fully supervised methods have been very successful in certain activities [65, 265, 266], they come with a caveat of requiring a large portion of the data to be labelled, and it is sometimes difficult or extremely time-consuming to obtain ground-truth labels for the dataset. Thus, it is desirable to develop learning algorithms that are able to work with less labelled data (*i.e.* weakly supervised) [267, 268].

Weak supervision in particular can be very useful in underwater fish monitoring, where the limited dataset size and the time- and cost-prohibitive nature of labelling limits achieving a useful dataset for developing effective, smart, and automated habitat monitoring tools and techniques. A number of works in literature have already used weak supervision for underwater fish habitat monitoring. For example, [269] proposed a segmentation model that can efficiently train on underwater fish images, not manually segmented for training, but only labelled with simple point-level supervision. This work demonstrated that in the marine monitoring context, weakly-supervised learning can effectively improve the accuracy and speed of model development with limited dataset sizes and limited labelling budget.

Active Learning

Active learning is a sub-field of ML and, more broadly, of AI. In active learning, the proposed algorithm is allowed to be ”inquisitive”, that is, it is allowed to pick the data to learn, which in theory means the algorithm can do more with less guidance, similar to weak supervision. Active learning systems are seeking to solve the constraint of labelling by posing a questionnaire in the context of unlabeled examples to be labelled by an oracle (e.g. a human annotator). In this manner, the goal of the active learner is to attain high precision by using as few labelled examples as possible, thus minimising the expense of acquiring labelled data; see Figure 3.5.

In many cases, the labels come for little or no cost, like the ”spam” label that is used to mark spam emails, or the five-star rating that a user could post for a movie on a social networking platform. Learning methods use these labels and scores to help screen your

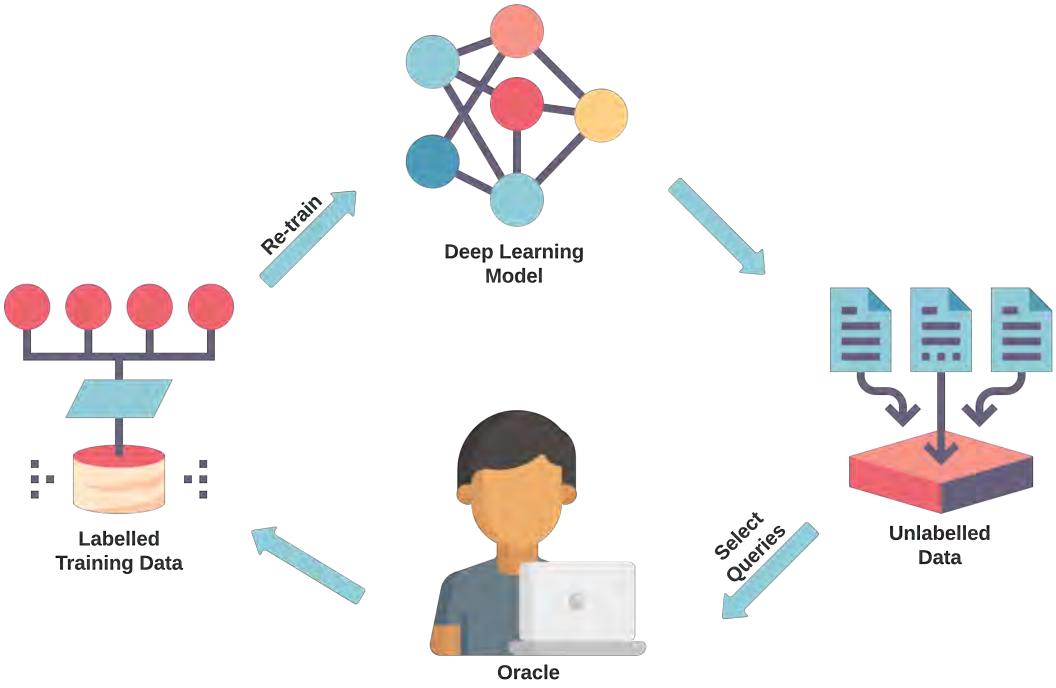

Figure 3.5: Schematic diagram of Active Learning

spam email and recommend movies that you might enjoy. In these cases, certain labels are given free of charge, but for more sophisticated supervised learning tasks, such as when you need to segment a fish in an underwater environment, this is not the case. For example, in [161] active learning has been used for the classification of species in underwater images from a fixed observatory. The authors proposed an active learning method that assigns taxonomic categories to single patches based on a set of human expert annotations, making use of cluster structures and relevance scores. This active learning method, compared to traditional sampling strategies, used significantly fewer manual labels to train a classifier.

Few-Shot Learning

The scarcity of rare species images in training datasets is one of the main limitations when addressing the automatic processing of wildlife images, especially in fish habitat monitoring. Such limitations lead researchers to explore few-shot learning.

Few-shot learning is another sub-field of ML. It is closely related to active learning since it aims to infer relationships between data from very few data samples. The central concept is how one can learn from a small number of examples and apply this knowledge to unlabelled data [270,271]. For example, you want to do animal identification in wildlife

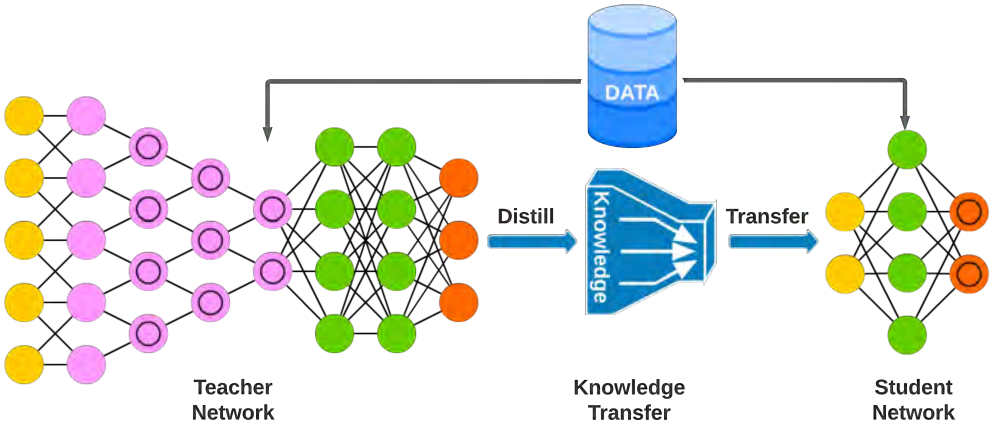

Figure 3.6: Schematic diagram of knowledge distillation

camera trap image datasets. However, since you have only a few labelled examples of rare species, with only a few images in training datasets, you cannot train your model to recognise these animals because you only have a few examples. In this case, few-shot learning can be used to learn how to use the previously learned classifier to recognise other features of objects on the image (e.g. shape) that might help you complete the task. However, training on these new features should be done in a few-shot manner [272, 273]. The idea is to have a pre-trained model trained on a much larger dataset of different species. Then, once a new species appears in the dataset of unlabelled images, you can use this pre-trained model to find similarities between the new image and those that are already in the dataset and label those that are similar to the target species.

In a pioneering study of using few-shot learning in processing underwater videos, [274] used it to discriminate 20 coral reef fish species with a range of training datasets from 1 image per class to 30 images per class. Few-shot object detection has been also used to localise wildlife using a camera trap in [275]. In another study, [276] proposed a data augmentation method that applies constraints on the mixture of foreground and background images based on species distributions. Therefore, after training a convolutional neural network for species classification, the model can localise a new image to a species with the help of the species distribution constraints in the mixture of foreground and background images. Similar techniques can be used in addressing the scarcity of sample data for rare marine species in underwater videos.

Adaptive Loss

The cross-entropy loss can be overwhelmed by the large class imbalance between foreground and background classes in the dataset during the training of dense detectors. This is because it is based on an implicit assumption of equal class priorities and does not differentiate between easy or hard examples. Therefore, [277] proposed to use a weighted cross-entropy loss, which assigns higher weights to the loss of hard samples and down-weight easy examples, thus focusing the training on hard negatives. The adaptive focal loss $FL(p_{\mathrm{t}})$ is derived from the entropy loss.

$$\mathrm{FL}\left(p_{\mathrm{t}}\right)=-\alpha_{\mathrm{t}}\left(1-p_{\mathrm{t}}\right)^{\gamma}\log\left(p_{\mathrm{t}}\right)$$

where α balances the importance of positive and negative examples, p_{t} is predicted probability, $(1-p_{\mathrm{t}})^{\gamma}$ is a modulating factor to the cross-entropy loss, and γ is a tunable focusing parameter. It has been shown in [277] that adaptive focal loss improves the accuracy compared to other losses for object detection on COCO test-dev [264].

In marine and fish habitat monitoring applications, it is very likely that strong class imbalance happens when datasets are being collected. This is mainly because the collected videos will have more examples of specific backgrounds such as coral reef, compared to various species of fish of interest. To address these issues, in addition to techniques such as adaptive loss mentioned above, other techniques developed for dealing with the problem of long-tailed distribution of training data can be explored and adopted. These include techniques such as those proposed in [278] where the authors proposed a class-balanced loss to re-weight loss inversely with the adequate number of samples per class, or by replacing the standard cross-entropy in [279] with label-distribution-aware margin loss.

3.6.4 Biodiversity Challenges

In a recent article, [272] have discussed some challenges beyond dataset limitation, focusing on biodiversity and how it can affect the deep learning-based automatic monitoring of marine and fish habitats through computer vision. Specifically, they consider the implications of three major universal rules of biodiversity, i.e. the distribution of species abundance, species rarity, and ecosystem openness [272]. The authors discuss how these rules bring about three main issues affecting the performance of deep learning algorithms for underwater monitoring. They also discuss promising solutions to these issues, some

of which were already discussed in subsection 3.6.3. Due to the importance of these issues and the challenges they pose to fish habitat monitoring, we briefly discuss them here. However, the reader is encouraged to refer to [272] for further details.

The first issue discussed is the imbalance of long-tail datasets, which is due to the abundance of some species in the collected videos and datasets, while some other groups may only be represented occasionally. Similarly, the second discussed issue is scarce data due to species rarity, which is a prominent biodiversity issue. Both the "long-tail datasets" and the "scarce data" issues, can cause a classifier to overfit the majority classes and fail to detect or predict the minority classes [278]. One way to tackle this issue is by data augmentation (see section 3.6.3) or Few-shot learning (see section 3.6.3). The other way is in the training algorithms itself by modifying the loss function with respect to dataset imbalances (see section 3.6.3).

The third challenge discussed is the "open world" issue that deals with an open ecosystem creatures. This results in the challenge of always having a new species that the "closed world" application is not trained on. This leads the model to misclassify the known species especially when the goal is to detect and predict marine species at sea. [272] discuss open-set learning as a way of solving such a problem. The objective of an open-set recognition model is to classify all samples belonging to the training dataset correctly while allowing it to ignore all samples of the novel classes [280].

3.7 Opportunities in Applications of DL to Underwater Fish Monitoring

New methodologies and strategies should be developed to advance DL models for various underwater visual monitoring applications, including fish monitoring, and to bring them closer to their terrestrial monitoring equivalents. In a previous study that was focused on the task of fish classification [1], we have discussed some of the future research opportunities including (i) utilising Spatio-temporal data to add space and time domain information to the current training algorithms that mainly learn fish images regardless of their spatial and/or temporal correlation; (ii) Developing efficient and compact DL models that can be deployed underwater for real-time parsing of the fish images at the collection edge; (iii) Combining image data from multiple collection platforms for improved multi-faceted learning; and (iv) Automated fish measurement and monitoring from underwater captured images. Figure 3.7 shows application scenarios for deep learning in underwater

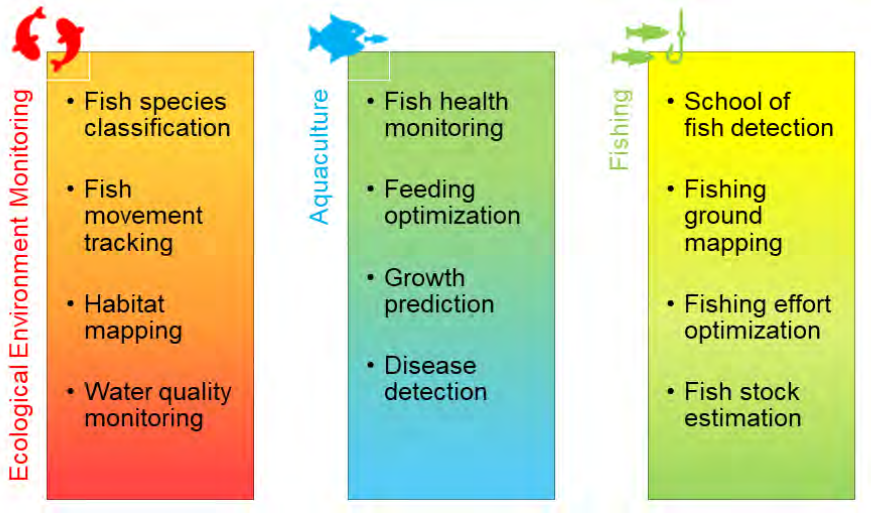

Figure 3.7: Application scenarios for deep learning in underwater fish monitoring, including ecological environment monitoring, aquaculture, and fishing. Deep learning can be used to classify fish species, track their movement patterns, monitor fish health, optimize feeding schedules, and identify schools of fish for more sustainable fishing practices.

fish monitoring, including ecological environment monitoring, aquaculture, and fishing, have been identified. Deep learning can be used to classify fish species, track their movement patterns, monitor fish health, optimize feeding schedules, and identify schools of fish for more sustainable fishing practices. In addition to the opportunities discussed in [1], further research areas could include (i) Developing DL models that can handle a wider range of image quality and visibility conditions, such as those encountered in murky or low-light environments; (ii) Combining visual monitoring with other sensor modalities such as acoustic sensing to improve detection and tracking accuracy; and (iii) Developing robust data labelling and annotation methods for large-scale training datasets, which can be difficult to obtain in underwater environments.

3.7.1 Knowledge Distillation for Underwater Embedded and Edge Processing

DL models used for fish monitoring applications are usually very large containing millions of parameters and requiring extensive computational power. To deploy these models on resource-limited devices and in resource-constrained environments such as undersea monitoring sites, different hardware-enabled compression techniques such as quantising and binarising DNN parameters [153] can be used, as discussed in [1]. Another method

that has seen a lot of interest and attention for compressing large-scale DL models is knowledge distillation.

Knowledge distillation is a technique for training a student (*i.e.* a small network) to emulate a teacher (*i.e.* ensemble of networks), as shown in Figure 3.6. The primary assumption is that in order to achieve a competitive or even superior performance, the student model should imitate the teacher model. The main issue is, however, transferring the knowledge from a large teacher to a smaller student. To that end, [281] proposed model compression as a way to transfer knowledge from a large model into a small model without sacrificing accuracy. In addition, several other model compression approaches have been developed, and the community has shown an increasing interest in knowledge distillation, due to its potentials [282–285].

A significant research opportunity lies in applying Knowledge distillation into embedded devices and underwater video processors to achieve online and more effective surveillance with high accuracy while using limited resources. This is particularly useful because of the limitations of transferring data from underwater sensors and cameras, and due to the challenging underwater communication in the Internet of Underwater Things [174].

3.7.2 Merging Image Data from Multiple Sources

As discussed in [1], to train more effective DNNs, multiple data collection platforms like Autonomous Underwater Vehicles (AUVs) or inhabited submarines can give varied visual data from the same monitoring subject. This can provide additional monitoring information, such as fish distribution patterns. Although it is straightforward to combine multiple data sources for training a DL network, several issues should be addressed in future research. These include possible preprocessing on part of data to make it compatible with the rest of the training dataset, class-wise weights (i.e. when you have an imbalanced dataset), and the number of outputs of a network. In addition, multiple training data sources, in particular, when using AUVs or submarines, incurs significant data collection and manual labelling cost, which is not always viable.

For this reason, some researchers have focused on learning from data with the least amount of human labelling. To reduce human-labelled data cost, several methods have been proposed to train models on data that are unlabeled [286] or only have pseudo-labels [287]. Future research can advance this further by developing faster and cheaper annotating tools for underwater fish images.

3.7.3 Prospective Research

Deep learning has proven to be an effective tool for analyzing and monitoring fish habitats and behaviour. However, there are still several areas where research is needed to further advance the use of DL in fish monitoring [1]. In this section, we discuss some prospective research directions that can increase the performance and usability of DL-based visual fish monitoring tasks.

1. Spatio-temporal data utilization: DL models mainly learn fish images regardless of their spatial and temporal correlation. Utilizing spatio-temporal data can add space and time domain information to the current training algorithms, leading to improved accuracy and robustness of the models. One potential approach is to use convolutional neural networks (CNNs) with 3D convolutions to learn both spatial and temporal features from video data. Another approach is to use recurrent neural networks (RNNs) to model temporal dependencies in sequential data, such as fish movement trajectories [4].

2. Efficient and compact DL models: To deploy DL models underwater for real-time parsing of fish images at the collection edge, compact and efficient models are needed. The current state-of-the-art DL models are often computationally expensive and require large amounts of memory. Research can focus on developing lightweight architectures that can be efficiently deployed on resource-constrained devices [174]. One approach is to use knowledge distillation techniques to transfer knowledge from a large pre-trained model to a smaller model while maintaining performance.

3. Multi-platform data fusion: Combining image data from multiple collection platforms, such as sonar and acoustic sensors, can improve the multi-faceted learning of DL models. However, integrating data from different sources poses several challenges, including differences in data quality and format. Developing effective techniques for data fusion, such as transfer learning and domain adaptation, can help to overcome these challenges and improve the performance of DL models for fish monitoring (see section 3.4.5).

4. Automated fish measurement and monitoring: Fish size and behaviour are important indicators of ecosystem health. Manual measurement and monitoring of fish can be time-consuming and expensive. DL models can automate this process by

extracting size and behaviour features from underwater captured images [6]. Research can focus on developing DL models that can accurately measure the fish size and identify behavioural patterns, such as swimming speed and direction.

In summary, prospective research directions can expand the capabilities and effectiveness of DL-based visual fish monitoring tasks. Utilizing spatio-temporal data, developing efficient and compact models, multi-platform data fusion, and automated fish measurement and monitoring are some of the areas that can lead to further advancements in the field.

3.8 Summary and Conclusion

The goal of this article was to provide researchers and practitioners with a summary of the contemporary applications of DL in underwater visual monitoring of fish, as well as to make it easier to apply DL to tackle real challenges in fish-related marine science.

DL has progressed as a technology capable of providing unprecedented benefits to various aspects of marine research and fish habitat monitoring. We envision a future where DL, complemented by many other advances in monitoring hardware and underwater communication technologies [174], is widely used in marine habitat monitoring for (1) data collection and feature extraction to improve the quality of automatic monitoring tools; and (2) to provide a reliable means of surveying fish habitats and understanding their dynamics. We expect that such a future will allow marine ecosystem researchers and practitioners to increase the efficiency of their monitoring efforts. To achieve this, we need concentrated and coordinated data collection, model development, and model deployment efforts. We also need transparent and reproducible research data and tools, which help us reach our target sooner.

Chapter 4

Transformer-based Self-Supervised Fish Segmentation in Underwater Videos

Underwater fish segmentation to estimate fish body measurements is still largely unsolved due to the complex underwater environment. Relying on fully-supervised segmentation models requires collecting per-pixel labels, which is time-consuming and prone to overfitting. Self-supervised learning methods can help avoid the requirement of large annotated training datasets, however, to be useful in real-world applications, they should achieve good segmentation quality. In this Chapter, the first research question is addressed. Specifically, we introduce a Transformer-based method that uses self-supervision for high-quality fish segmentation. Our proposed model is trained on videos ***without any annotations*** to perform fish segmentation in underwater videos taken in situ in the wild. We show that when trained on a set of underwater videos from one dataset, the proposed model surpasses previous Convolutional Neural Network (CNN)-based and Transformer-based self-supervised methods and achieves performance relatively close to supervised methods on two new unseen underwater video datasets. This demonstrates the great generalisability of our model and the fact that it does not need a pre-trained model. In addition, we show that, due to its dense representation learning, our model is compute-efficient. We provide quantitative and qualitative results that demonstrate our model's significant capabilities.

4.1 Introduction

Fish segmentation is an important yet challenging task that plays a critical role in marine and aquaculture applications such as fish body measurements, fish breeding, fish counting, and fishing-related activities. The goal of fish segmentation in underwater images and videos is to produce a pixel-wise mask for each fish in the video/image. This mask can then be used to perform subsequent body measurements like length and width of fish, or extract its body shape. However, the underwater environment usually bring challenges such as blurry images, cluttered background, and similarity between fish and its surrounding environment, which make the process of underwater fish segmentation extremely difficult.

Previous methods for underwater fish segmentation [9, 125, 288, 289] mainly relied on fully-supervised models that require human-generated segmentation masks for training. These trained models usually perform well for a specific, small set of datasets, but their performance drops when applied to other unseen datasets, e.g. from other underwater fish habitats. In addition, it is usually difficult to obtain large, in-the-wild underwater datasets, making it more challenging to produce models that generalise well.

To improve the generalisability of segmentation models and resolve the issue of limited access to large-scale underwater videos, self-supervised video segmentation (*aka*. dense tracking) can be used. However, when applied to underwater scenarios, these self-supervised models face additional challenges compared to their terrestrial counterparts, due to the limited underwater optical view. Solving these challenges would help develop new underwater optical/acoustic imaging or autonomous robot navigation systems.

To that end, the primary motivation of this work is to address the lack of an efficient underwater fish segmentation method with good generalisability, which is important for various applications such as marine biology, ocean conservation, and underwater robotics. We are also motivated by the fact that self-supervised learning provides the advantage of not requiring manually annotated data for training, which is highly beneficial in the context of underwater video object segmentation, where the manual annotation is time-consuming and costly. Our proposed method is also motivated and inspired by the strong performance of the self-attention mechanisms of the Transformer models, which have not been previously applied to underwater segmentation tasks.

Architectures based on Transformer models, such as Vision Transformer (ViT) [290], have been shown to outperform standard Convolutional Neural Networks (CNNs) in many tasks, especially for large datasets. Our method uses a contrastive formulation and a self-

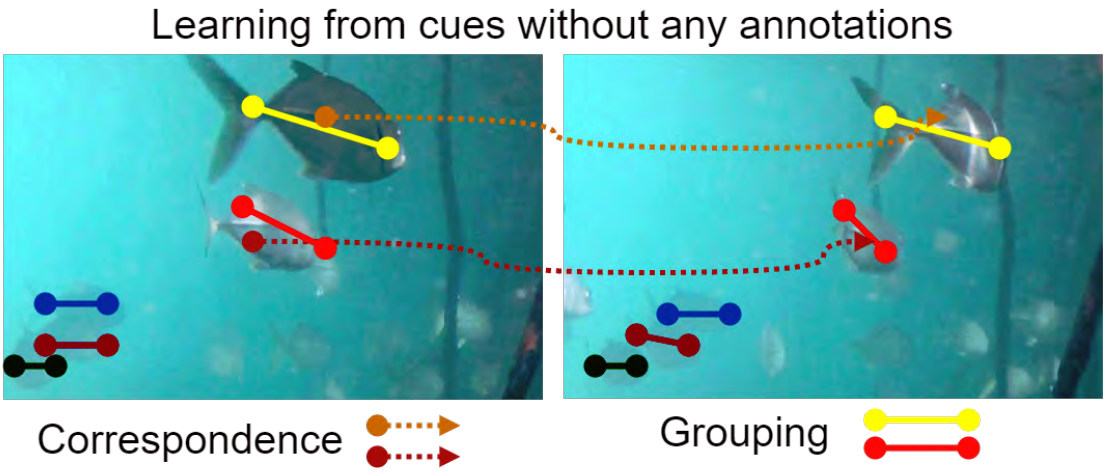

Figure 4.1: The natural visual artefact dynamics provide important cues about the composition of scenes, and how they change.

training objective to generate pseudo labels based on a transformed view of the original video sequence, and learns to assign the features in the original view consistently with the transformed view. The self-supervised loss also helps disambiguate the anchor points spatially and between independent video sequences, ensuring their transformation equivariance. Overall, the goal of the method is to learn a representation of the frames that is robust to different underwater scenes and can effectively segment objects in underwater video.

Unlike previous underwater fish segmentation methods , our proposed method can achieve high-quality underwater fish segmentation without a pre-trained model or any annotations. Our work can also be seen as a specific instance of the more general segmentation methods proposed in [12], in the domain of underwater fish video segmentation.

The research contributions are as follows:

- Introduction of a Transformer-based method that uses self-supervision for high-quality fish segmentation.
- The proposed model is trained on videos without any annotations and surpasses previous CNN-based and Transformer-based self-supervised methods.
- The model achieves performance relatively close to supervised methods on two new unseen underwater video datasets, demonstrating its great generalizability.
- The model is compute-efficient due to its dense representation learning.

The rest of the paper is organized as follows. Sec. 4.2 covers related works and provides background information on the novel aspects of our work. Our model's framework is described in detail in Sec. 4.3. Sec. 4.4 presents our method for training and evaluating our self-supervised learning model and the experimental setup and results, while detailed discussions of our results are presented in Sec. 4.5. Finally, Sec. 4.6 concludes our paper.

4.2 Related Work

Video object segmentation (VOS) [291] without supervision has been an active area of research in recent years. Several researchers [292] have exploited spatiotemporal information in videos to learn dense feature representations. In this section, we briefly review the research domains most relevant to our work.

Supervised And Unsupervised Learning. The process of learning can be either supervised or unsupervised. In the case of supervised learning [1], we have a dataset, in which each datum has a corresponding label. Therefore, the learning algorithm will be trained in such a way that it assigns the right label to the data and does not deviate from the specified label. In unsupervised learning [293] on the other hand, the dataset does not include corresponding labels. Unsupervised learning tries to find the intrinsic structure in the data. For instance, previous methods [294] have exploited the spatiotemporal ordering of video frames to extract supervisory signals.

In this work, we focus on unsupervised learning. Our proposed method is a self-supervised video object segmentation model trained on videos without any annotations. Therefore, there are no supervising signals (labels) available for the learning process. Hence, there is no explicit correspondence between a video and a label.

Representation learning is a class of machine learning approaches that model knowledge or representations about data (i.e. decompose training samples into feature representations) [295]. Representations can be used to learn rules for classification or to represent objects that can be used for a variety of tasks such as visual object recognition, semantic understanding, and other tasks. Learning spatiotemporal representations from videos has been extensively researched [296]. However, these studies mainly learn global feature representations, not dense representations. Pinheiro *et al.* [297] proposed a view-agnostic model for dense representations of static scenes through pixel-level contrastive learning. In contrast to [297], which is limited to image sets of static scenes, our model learns dense representations of dynamic scenes from videos.

Contrastive Learning is a popular form of self-supervised learning [298]. It assumes

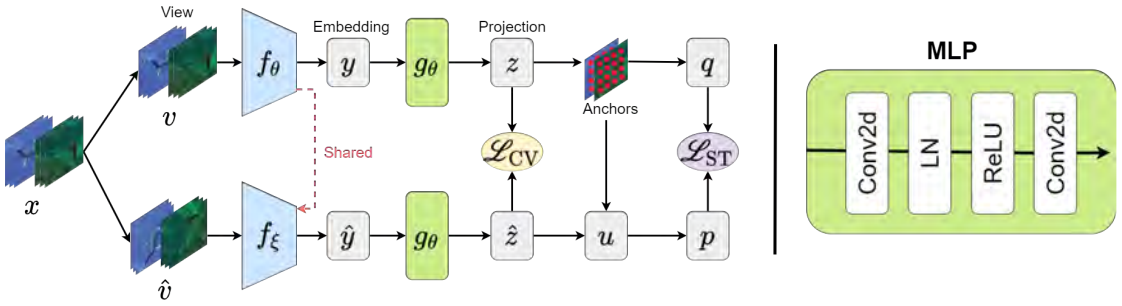

Figure 4.2: Our proposed framework consists of a single Transformer-based feature extractor that processes video sequences. Given a batch of unlabeled video sequences x, two batches of different views v and $\hat{v}$ are produced and are then encoded into embeddings y and $\hat{y}$ through the main branch f_θ and the second regularising branch f_ξ, respectively. The embeddings are fed to a multilayer perceptron (MLP) g_θ to produce the projections z and $\hat{z}$ to compute the cross-view consistency loss $\mathcal{L}_{\text{CV}}$. The self-training loss $\mathcal{L}_{\text{ST}}$ learns space-time embeddings between the anchors q and pseudo labels p (arg max of u, affinities of $\hat{z}$ *w. r. t.* anchors.). The two branches are identical in architecture with shared weights. The encoders f are CoaT Transformer [33] backbones.

visual features are invariant under a certain set of data that has two or more views and learns the representations to distinguish each view from the others. Contrastive learning approaches have also been applied to many visual classification problems in which one learns the representations invariant to scale and rotation [299]. Contrastive learning may also be thought of as a classification technique that classifies data by maximising feature resemblances between an image and its augmented instance, while minimising the resemblance between negative samples. For example, SimCLR [300] learns generic representations of images from an unlabeled dataset. Momentum Contrast (MoCo) [301] also exploits the negative samples on high-dimensional continuous inputs, such as images, to build a large and consistent dictionary for learning visual representations by keeping a memory bank of negative samples.

In contrast to manually augmenting the still images as in the existing contrastive methods [301], we utilize the natural visual artefact changes in a natural scene directly from video data, *i. e.* temporally adjacent frames in videos.

Correspondence Learning aims at training a deep network by automatically predicting correspondences between image pairs [302]. In this way, the network can be trained with a limited number of image pairs, which eliminates the need for annotations. For instance, Fig. 4.1 provides an example of a spatiotemporally correlated image pair in underwater videos). When the input is a video stream, this approach is particularly useful

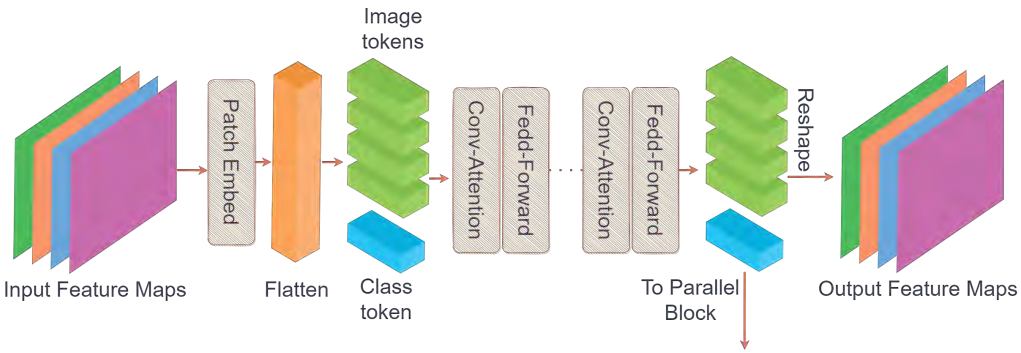

Figure 4.3: Schematic graph of the serial block in CoaT Transformer [33]. Input feature maps are first down-sampled by a patch embedding layer and then flatten the reduced feature maps into a sequence of image tokens. Multiple Conv-Attention and Feed-Forward layers process the tokenized features, along with a class token (a vector to achieve image classification).

and has recently been shown to yield interesting results [13]. Jabri *et al.* [11] used the contrastive random walk to learn a representation for visual correspondence from raw video. Araslanov *et al.* [12] took a step further by learning dense representations in a fully convolutional manner.

In contrast to [11] that uses only intra-video self-supervision, our work is similar to [303] by using both inter- and intra-video level consistency to learn more discriminatory feature embeddings.

Vision Transformers (ViT). Transformers in machine learning are composed of multiple self-attention layers. They are primarily used in natural language processing and often achieve impressive results [304]. For many computer vision applications, CNNs have long been the gold standard [8], yet the convolution operator makes modelling long-range interactions difficult. For this reason and due to their success in NLP, Dosovitskiy *et al.* [290] introduced Vision Transformer (ViT) by applying self-attention mechanisms to image patches to generate features for image classification. This approach obtained state-of-the-art results on ImageNet.

Despite these promising results, there is still room for improvement in terms of segmentation quality and generalizability to different underwater environments and fish species.

In our work, we explore attention over all possible patches in an image and the entire image at once. We apply transformer-like architectures on patches and entire images. For patches, we use 16×16 grids, so that the resulting transformations can be used to generate an entire image. However, larger or smaller grids can also be used.

4.3 Method

An overview of our model and its training procedure is presented in Fig. 4.2. Given a batch of unlabeled video sequences, two batches of different views are produced and are then encoded into embeddings through the main branch and the second regularising branch. The two branches are identical in architecture with shared weights. The encoders are CoaT Transformer [33] backbones. The embeddings are fed to a multilayer perceptron (MLP) to produce the projections to compute a cross-view consistency loss, while a self-training loss helps learn space-time embeddings between introduced anchors and pseudo labels, which are explained in details below.

4.3.1 CoaT Transformer

Our feature encoder backbone is Co-scale conv-attentional image Transformers (CoaT) [33]. CoaT is composed of two submodules: (1) a conv-attentional image transformer (CAIT) module and (2) a co-scale feature attention network (CFAN) module. The CAIT module uses a spatial transformer network and convolutional operations to produce a co-scale feature pyramid from a single input image, and to realize relative position embeddings with convolutions in the factorized attention mechanism. The CFAN network operates on top of CAIT-produced feature pyramid representations and dynamically selects informative image parts to make decisions on what to encode and what to ignore for scene understanding, allowing us to model spatial and semantic relationships at multiple scales.

CFAN is composed of two sub-modules, a serial and a parallel block, which introduce fine-to-coarse, coarse-to-fine, and cross-scale information into image transformers. The *serial block* (shown in Fig. 4.3) models image representations at a downsized resolution, while a *parallel block* realizes a co-scale mechanism. Given an input image $I \in \mathbb{R}^{H \times W \times C}$, each *serial block* down-samples the image features into lower resolution, resulting in a sequence of four resolutions:

$$F_1 \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times C_1},$$

$$F_2 \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times C_2},$$

$$F_3 \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times C_3},$$

$$F_4 \in \mathbb{R}^{\frac{H}{32} \times \frac{W}{32} \times C_4}.$$

Since the CFAN module produces multi-scale feature attention maps from a single image, it is a more computationally efficient and scalable method than the existing multi-resolution encoder-decoder frameworks. In addition, since CFAN takes input in the form of feature pyramid representation and produces a pyramid of feature attention maps, it is more flexible than existing multi-resolution architectures that operate on fixed-sized feature pyramids.

We, therefore, use CoaT as a feature encoder in the two branches of our model, main and regularising. The main and regularising branches process two copies of the input frame batch, each of which includes the identical collection of video sequences. We feed the augmented version of each frame to the *regularising branch*, while the *main branch* receives the original video frames (as shown in Fig. 4.2). For augmentation, we extract random cropping and flipping, as described in Sec. 4.4.2. The regularising branch's purpose is to avoid the degenerate solutions that make the network encode positional cues into a degenerate feature representation, as previously reported in [11].

4.3.2 Multilayer Perceptron (MLP)

We pass the output from the feature encoder through a multilayer perceptron (MLP) to produce feature embeddings and to reduce the feature dimensionality from 512 to 128. The multilayer perceptron (MLP) code implementation in PyTorch-like style is shown in algorithm 1. The MLP consists of two standard *Conv2d* layers. The first layer is followed by Layer Normalisation [305] and ReLU.

Algorithm 1: Multilayer Perceptron (MLP), PyTorch-like

```
import torch.nn as nn

class MLP(nn.Sequential):

    def __init__(self, n_in, n_out):
        super().__init__()

        self.add_module("conv1", nn.Conv2d(n_in, n_in, 1, 1))
        self.add_module("ln1", nn.LayerNorm(n_in))
        self.add_module("relu", nn.ReLU(True))
        self.add_module("conv2", nn.Conv2d(n_in, n_out, 1, 1))
```

4.3.3 Anchor Sampling

To improve training efficiency and computational footprint [11], we obtain (n^2) z-dimensional feature embeddings by defining a spatially invariant grid of size $n \times n$ on the feature tensor from the main branch z, and select one sample per grid cell (*i. e. anchors* k). This will make the anchors spatially distinct and cover the full feature embeddings. We then share these anchors with the regularising branch. Rather than computing pairwise distances between every feature vector in the batch, we compute the cosine similarities between the embeddings of the anchors k and the current features z by:

$$q_{i,j} = \frac{\exp(z_i \cdot k_j / \tau)}{\sum_l \exp(z_i \cdot k_l / \tau)} \tag{4.1}$$

where $\tau \in \mathbb{R}^+$ is a scalar temperature hyperparameter, z and k are features from the main branch and the anchors, respectively, l indexes batch samples and i, j index the vector dimension.

For the regularising branch features, we select only the predominant anchors [12] to compute the cosine similarities as follows:

$$p_i = \underset{j \in \mathcal{N}(i)}{\arg\max} \frac{\exp(\hat{z}_i \cdot k_j / \tau)}{\sum_l \exp(\hat{z}_i \cdot k_l / \tau)} \ , \tag{4.2}$$

where $\mathcal{N}(i)$ is the index set of the anchors that stem from the same video clip as the feature vector with index i, and $\hat{z}$ and k are features from the regularising branches and the anchors, respectively.

Note that, in Eq. (4.1) we extract features from multiple videos in the training batch, however, in Eq. (4.2) we extract features from the same video sequence only. This will help our framework to simultaneously learn intra-video (within a single video clip), and inter-video (between video clips) feature embeddings to preserve the fine-grained correspondence associations as well as instance-level feature discrimination [306].

4.3.4 Loss Function

The goal of training our framework is to learn representation as similarity across views (see Fig. 4.4) by computing pairwise affinities between features from the model's two branches and minimising the distance of the features extracted from the other temporally close frames to the anchors. Therefore, the overall loss of the learning algorithm is given by the following equations:

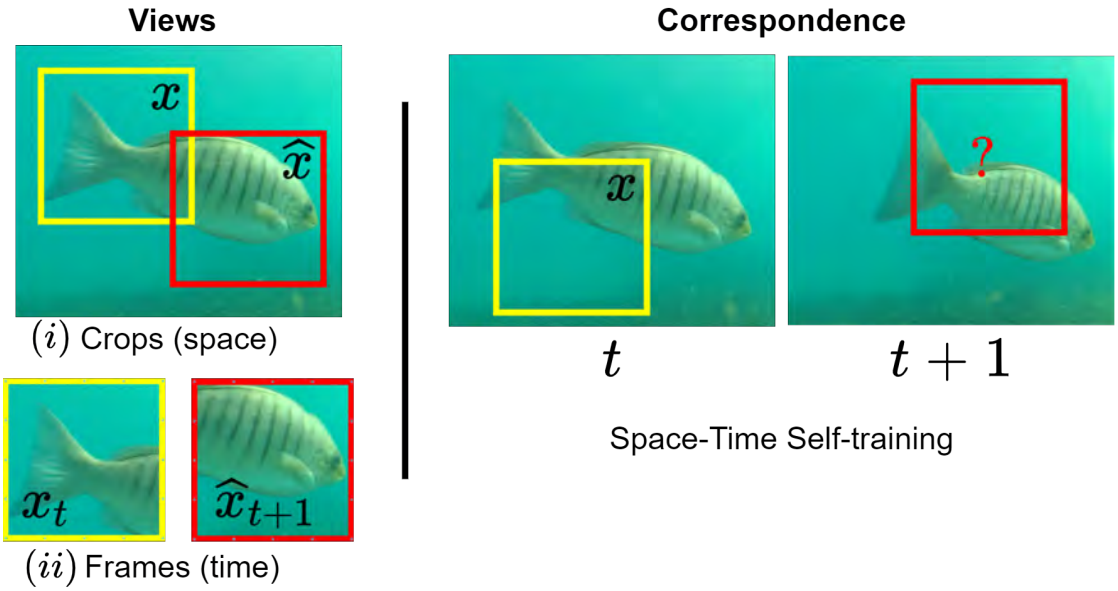

Figure 4.4: Representation Learning as similarity across views by discriminating features *(i)* spatially within individual frames and *(ii)* temporally, to represent each frame in a video sequence in terms of the same feature set.

Cross-view consistency

We build the input to the regularising branch by augmenting the original video frames to generate a random similarity transformation. The corresponding change in the features output, *i. e.* segmentation, should be the same regardless of randomly flipping or scaling the input frame. By using the cross-view consistency loss [12] in Eq. (4.3) we explicitly facilitate this property.

$$\mathcal{L}_{CV} = -\sum\nolimits_{i\in\mathcal{R}} \log \frac{\exp(z_i \cdot \hat{z}_i/\tau)}{\sum_{l\neq i}\exp(z_i \cdot \hat{z}_l/\tau)}, \tag{4.3}$$

where $\mathcal{R}$ is the index set of the features extracted from the reference frames; $\tau \in \mathbb{R}^+$ is a scalar temperature hyperparameter; and z and $\hat{z}$ are features from the main and the regularising branches, respectively.

Since z and $\hat{z}$ are spatially coordinated, this association distinguishes the cosine similarity between the corresponding features *w. r. t.* non-corresponding pairs.

Space-time self-training

After generating pseudo labels from the predominant anchor index for each feature from the regularising branch, we use space-time self-training loss [12], shown in Eq. (4.4), to

minimise the distance between the features extracted from the original view q defined by Eq. (4.1), and pseudo labels p, based on Eq. (4.2).

$$\mathcal{L}_{ST} = -\sum\nolimits_{i \notin \mathcal{R}} \log \mathcal{T}(q_i, p_i) \ , \tag{4.4}$$

where $\mathcal{R}$ is the index set of the features extracted from the reference frames, while $\mathcal{T}(\cdot)$ is random similarity transform to spatially aligns q and p after random cropping and flipping.

Since the anchors are a subset of features sampled spatially and temporarily from the video frames within the same view, this loss minimises the feature distance to the anchors and stimulates an increased cosine similarity of the features to the anchors and a decreased cosine similarity between the anchors themselves.

The final training objective is to minimise the combination of the above loss functions:

$$\mathcal{L} = \mathcal{L}_{CV} + \lambda \mathcal{L}_{ST}, \tag{4.5}$$

where λ is a hyperparameter that weights its contribution to the total loss.

4.3.5 Label Propagation

We use label propagation to predict semantic labels for all video clip frames from the initial ground-truth label only. Label propagation is the task of classifying each individual pixel in the frames of a video given only ground truth for the first frame. Following previous work [12], we employ the representation as a similarity function for k-Nearest Neighbour (KNN) prediction.

Algorithm 2 illustrates the label propagation we use in our work. We employ *context* embeddings and masks acquired from previous frames to forecast the mask m_t for the current time-step t. We use the output from the *CoaT Transformer* [33] to obtain the embedding for frame t. Then, we compute the cosine similarity of embedding e_t *w. r. t.* all embeddings in context $\mathcal{E}$, commonly used in correlation layers of optical flow networks [307]. Next, we compute local attention in a single operation by *kNN-Softmax*. Finally, we update the oldest entries by replacing them with m_t and e_t to the mask $\mathcal{M}$ and embedding contexts $\mathcal{E}$. For the remaining frames in the video clip, we repeat the same process. Bilinear interpolation is used to bring the final object masks back to their original resolution.

Algorithm 2: Label Propagation

Input: Embeddings $\mathcal{E}$ and mask $\mathcal{M}$ from the first frame.
Output: Mask m_t prediction for timestep t.

```
for t and frame in (frames) do
    Computing embeddings e_t at timestep t;
    Computing local spatial correlation between E and e_t;
    Computing softmax between K-Nearest Neighbors;
    Mask m_t prediction for timestep t;
    updating E and M;
```

Figure 4.5: Sample image from each of the three utilised datasets. From left: DeepFish [17], Seagrass [10] , and YouTube-VOS [16]

4.4 Experiments

We present the method of training and evaluating our self-supervised learning model for underwater video segmentation.We provide quantitative and qualitative results that demonstrate our model's generalization capabilities to a range of different underwater habitats.

4.4.1 Datasets

We performed experiments using three publicly available datasets, i.e. DeepFish [17], Seagrass [10] , and YouTube-VOS [16]. Fig. 4.5 demonstrates a sample image from each dataset.

DeepFish [17] consists of a large number of videos collected for 20 different habitats in remote coastal marine environments of tropical Australia. The video clips were captured in full HD resolution (1920×1080 pixels) using a digital camera. In total, the number of video frames taken is about $40k$

Seagrass [10] is comprised of annotated footage of *Girella tricuspidata* in two estuary systems in south-east Queensland, Australia. The raw data was obtained using submerged action cameras (HD $1080p$). The dataset includes 9429 annotations and 4280 video frames. Each annotation includes segmentation masks that outline the species as a

polygon.

YouTube-VOS [16] is a video object segmentation dataset that contains 4453 YouTube video clips and 94 object categories. The videos have pixel-level ground truth annotations for every $5th$ frame ($6fps$). For a fair comparison, we extracted only the videos that contained *fish*, which include 130 videos and 4349 video frames in total.

We independently train our feature extractor on the DeepFish [17] dataset and evaluate it on Seagrass [10] and YouTube-VOS [16].

4.4.2 Data Augmentation

In addition to natural variances in the video sequences, we use similarity transformations to augment the training data (random cropping and flipping only), see Fig. 4.4. The reason for using these extra augmentations is to augment the same input video to feed to the second regularising branch to produce pseudo labels, see Sec. 4.3.1.

We also experimented with several spatial and pixel-level augmentations, *e. g.* sheering, rotations, RGB-Shift, and colour jittering. However, we did not observe a notable change in accuracy. These augmentation methods are computationally expensive, because both rotation and sheering require image padding, which needs to be removed afterwards. Therefore, the video sequences were augmented with random flips and cropping only.

4.4.3 Implementation Details

Model training

We use a Transformer-based feature encoder as the backbone network for our feature extractor (see Sec. 4.3.1). As a baseline, we adopt the ResNet-18 feature encoder [226] as used in [11]. Similar to [303], we also remove the strides in the *res3* and *res4* blocks from the ResNet-18 architecture. Both our proposed Transformer-based and baseline models' weights were randomly initialised.

Our models were trained with an input resolution of 256×256 pixels. We scale the lowest side of the video frames to 256 and then extract random crops of size 256×256. We sample two video sets, $B = 2$ (of size $T = 5$ frames), therefore, $B \times T = 2 \times 5 = 10$ frames are used per forward pass.

We found that for this problem set, a learning rate of 1×10^{-3} works the best. It took around 300 epochs for all models to train on this problem. Our networks were trained on a Linux host with a single NVidia GeForce RTX 2080 Ti GPU with 11 GB of memory, using

Pytorch framework [308]. We used Adam optimiser [204] with $\beta_1 = 0.5$, $\beta_2 = 0.999$, and $\epsilon = 1.0 \times 10^{-08}$. We applied the same hyperparameter configuration for all of the models. However, the optimum model configuration will depend on the application, hence, these results are not intended to represent a complete search of model configurations. Our training loop is shown in algorithm 3.

Algorithm 3: Main Training Loop

Input: Unlabeled video sequences.
Output: Trained weights for the backbone network.

```
for each mini-batch do
    Extract deep features of the video frames;
    Regularising branch produces pseudo labels;
    for each video in the mini-batch do
        // Transformer-based encoder
        Extract feature embeddings (anchors) k;
        Compute affinity to anchors q (Eq. (4.1));
        Compute pseudo labels q (Eq. (4.2));
        // Loss Computation
        Compute Cross-view consistency L_CV (Eq. (4.3));
        Compute Space-time self-training L_ST (Eq. (4.4));
        Compute total loss L (Eq. (4.5));
    Back-propagate all the losses in this mini-batch;
```

Inference

At the inference time, we compute dense correspondences for video propagation using the learned encoder's representation. The encoder's representation is the trained weights for the backbone network (see Sec. 4.3.1 and Algorithm 3). We predict the whole video frames segmentation masks using Label Propagation (Sec. 4.3.5). Given the initial ground-truth segmentation mask of the first frame, we label propagate the rest of the frames in the video without the need for the rest of ground-truth annotations. The labels are propagated in the feature space. The labels in the first frame are one-hot vectors, whereas the labels propagated are Softmax distributions.

4.4.4 Evaluation Metrics

In the context of video segmentation, there are two popular evaluation metrics that can be used to assess the performance of a self-supervised learning model:

1) The Jaccard's index ($\mathcal{J}$): This metric measures the overlap between the predicted segmentation mask and the ground truth mask, and is calculated as the ratio of the intersection of the two masks to their union. A higher $\mathcal{J}$ score indicates a better segmentation performance. In mathematical notation, the Jaccard index can be represented as follows: $\mathcal{J}(A, B) = \frac{|A \cap B|}{|A \cup B|}$, where A and B are the predicted segmentation mask and the ground truth mask, respectively.

2) Dice coefficient ($\mathcal{F}$): It is a measure of the overlap between the predicted and ground truth segmentations, and is calculated as the ratio of the intersection of the two segmentations to their sum. The Dice score ranges from 0 to 1, with a value of 1 indicating perfect overlap between the predicted and ground truth segmentations, and a value of 0 indicating no overlap. It is calculated as $\mathcal{F} = \frac{2|A \cap B|}{|A|+|B|}$, where A and B are the predicted segmentation mask and the ground truth mask, respectively.

Here, we report the mean average of $\mathcal{J}\&\mathcal{F}$, and the mean and recall of both $\mathcal{J}_m$ and $\mathcal{F}_m$, with an IoU threshold of 0.5.

4.4.5 Compared Methods

We evaluated the performance of our method against **five** other methods for self-supervised video object segmentation and one fully-supervised method. All of these methods used the same feature extraction network, ResNet18 CNN [12]. The self-supervised methods are: CRW [11], DenseFlow [12], MAST [13], Colorize [14], CorrFlow [15].

The Fully supervised fully convolutional neural network (FCN) method is based on the FCN8 architecture developed by Shelhamer *et al.* [232]. It uses the true per-pixel class labels to fully supervise the training process. The FCN method is efficient and can handle imbalanced datasets, where the number of pixels belonging to a certain class is much smaller than the number of pixels belonging to other classes. In our case, the number of pixels corresponding to the fish is much smaller than the number of pixels corresponding to the background. The comparison with the fully-supervised method aims to demonstrate the advantages of our trainable self-supervised deep-learning-based method.

For fair comparisons with our model, we used the authors' code and training approach for these self-supervised methods. To guarantee the experiment's objectivity, we trained the five methods on DeepFish [17] and applied the author-provided model and network training parameters.

4.4.6 Quantitative Comparisons

The comparison results for Seagrass [10] and YouTube-VOS [16] are summarized in Table 4.1 and Table 4.2, respectively. The best results for each metric are in ***bold***.

Compared to the second highest method [12], our approach reaches a higher $\mathcal{J}\&\mathcal{F}_m$ score by 4.5% and 3.1% on Seagrass and YouTube-VOS, respectively. This is mainly caused by using self-attention mechanisms that can extract high-level spatial features for segmenting long-range video sequences. Moreover, the self-attention layers can be used to model the dependency among multiple temporal steps, which is helpful for the segmentation of objects having large motions.

We also compared our method against a fully supervised method. Fully supervised video object segmentation involves the identification and segmentation of a specific object within a video sequence, where the object of interest has been predetermined and the model has been trained on a comprehensive dataset of annotated video frames. To train a model for this task, it is necessary to acquire a collection of annotated video frames, comprising a significant number of frames that have been manually labelled to show the pixels corresponding to the targeted object.

Manually segmenting the objects of interest in each video frame requires a high annotation budget. An annotation budget refers to the amount of time and resources that are allocated for the process of annotating data, which involves labelling and categorizing data in a consistent and structured way. In the context of segmenting a single fish in an image, the annotation budget would depend on various factors such as the complexity of the task, the number of images that need to be annotated, and the resources available for the annotation process. According to Saleh *et al.* [17], segmenting a single fish in an image took about 2 minutes. Given that, fully supervised training of a single video that contains only 100 frames would cost 200 minutes.

We report the FCN for semantic segmentation [232] as a fully-supervised learning method in Table 4.1 and Table 4.2. The fully-supervised method has a higher $\mathcal{J}\&\mathcal{F}_m$ score on Seagrass and YouTube-VOS. However, our method does not require annotated frames. Furthermore, because a self-supervised approach does not require an annotation budget, the annotation budget can be spent on other tasks. By removing the annotation bottleneck, our method aims to reduce the cost of the process of annotating data. Furthermore, a reduced annotation budget might be a more feasible approach in situations where the resources are limited. For example, when a large number of objects are present in a video, manual annotation would be prohibitively expensive. The reduction in the annotation budget would free up resources and make the task more feasible.

Table 4.1: Performance Comparison on **Seagrass** [10] dataset between our model and five state-of-the-art models (CRW [11] DenseFlow [12] MAST [13] Colorize [14] CorrFlow [15]).

Method	Backbone	$\mathcal{J}\&\mathcal{F}$(Mean) ↑	$\mathcal{J}$(Mean) ↑	$\mathcal{J}$(Recall) ↑	$\mathcal{F}$(Mean) ↑	$\mathcal{F}$(Recall) ↑
CRW [11]	ResNet-18	43.2	38.9	40.4	46.2	50.8
DenseFlow [12]	ResNet-18	45.5	40.2	41.0	50.7	54.7
MAST [13]	ResNet-18	40.3	37.1	38.7	43.8	48.5
Colorize [14]	ResNet-18	34.9	34.5	35.1	40.8	47.9
CorrFlow [15]	ResNet-18	39.4	36.8	36.9	42.7	47.2
Ours	Transformer	**50.0**	**41.5**	**43.3**	**58.1**	**65.4**
Fully-supervised [232]	ResNet-18	64.7	52.4	55.7	71.4	79.1

Table 4.2: Performance Comparison on **YouTube-VOS** [16] dataset between our model and five state-of-the-art models (CRW [11] DenseFlow [12] MAST [13] Colorize [14] CorrFlow [15]).

Method	Backbone	$\mathcal{J}\&\mathcal{F}$(Mean) ↑	$\mathcal{J}$(Mean) ↑	$\mathcal{J}$(Recall) ↑	$\mathcal{F}$(Mean) ↑	$\mathcal{F}$(Recall) ↑
CRW [11]	ResNet-18	59.9	58.6	71.5	57.8	68.7
DenseFlow [12]	ResNet-18	60.2	60.9	72.7	59.5	**70.0**
MAST [13]	ResNet-18	57.4	57.9	68.1	56.9	65.2
Colorize [14]	ResNet-18	53.7	54.1	65.9	55.4	64.8
CorrFlow [15]	ResNet-18	56.4	55.9	66.7	54.3	64.8
Ours	Transformer	**63.3**	**63.9**	**74.0**	**62.7**	69.6
Fully-supervised [232]	ResNet-18	79.3	78.2	89.3	70.5	83.7

4.4.7 Qualitative Results

To visually inspect the segmentation results to ensure that the model is correctly identifying the desired objects in the image, we compared the performance of our model with a CNN-based encoder baseline [12] model on the YouTube-VOS (rows 1 and 4) and and Seagrass (rows 2-3) datasets in Fig. 4.6. Our model was able to effectively distinguish between objects and backgrounds and handle occlusions better than the baseline model, as shown in the qualitative results. An apparent advantage of our model shown in the bottom panel is its ability to segment multiple and overlapping fish, in the scene, where its CNN counterpart fails. Fig. 4.6 also shows that our model is stable and can effectively locate

the fish in long videos despite complex scenes. Here, our model managed to segment up to frame number 80 based on the first frame. The CNN-based method, on the other hand, has the problem of failing to predict the segmentation in some situations when there is vagueness between the foreground object and the background, or when there are complex transformations in the videos. Whereas our method shows a strong ability to differentiate pixels with similar intensities. Furthermore, these results show that our proposed method can work on datasets with very small objects (see Videos 2 and 3 in Fig. 4.6).

To further demonstrate the efficacy of our method, Fig. 4.7 and Fig. 4.8 show a comparison between our model and five state-of-the-art self-supervised models (CRW [11], DenseFlow [12], MAST [13], Colorize [14], CorrFlow [15]) applied to the YouTube-VOS [16], and Seagrass [10] datasets.

4.4.8 Ablation Study

To further investigate the effect of our proposed transformer-based video segmentation framework, we performed an ablation study. In this study, we compared the baseline CNN model (no transformers) [12] and our Transformer-based feature encoder, with different types of transformers. For these comparisons, we used different configurations of video features, Transformers, and MPL layers, to find out which combination results in the best performance. We report only results with best configurations.

Table 4.3 reports the segmentation accuracy for four different models in terms of $\mathcal{J}\&\mathcal{F}_m$ metric, in addition to the base-line CNN [12] and our proposed models.In this Table, the second line is the Fast Fourier Convolution model in [309]. The third line refers to the Transformer for Semantic Segmentation introduced in [310], while the fourth line reports the MetaFormer-based architecture called PoolFormer [311]. The fifth line is for Cross-Covariance Image Transformer (XCiT) [312]. Based on the evaluation in Table 4.3, our Transformer-based model significantly outperforms the baseline and other meta-architecture methods in the context of both datasets.

In addition to the ablation studies presented in this chapter, we recognize the potential value of conducting further studies on various hyper-parameters of our proposed model. While these additional studies are beyond the scope of our current research, we plan to explore them in future work. For instance, we believe that conducting an ablation study on the cross-view loss parameter (λ) could reveal its impact on the performance of our model. Similarly, analyzing the effects of different numbers of sampled videos (B) and frames per video (T) could provide insights into the optimal settings for these parameters. Furthermore, investigating the impact of varying the size of the sampling grid

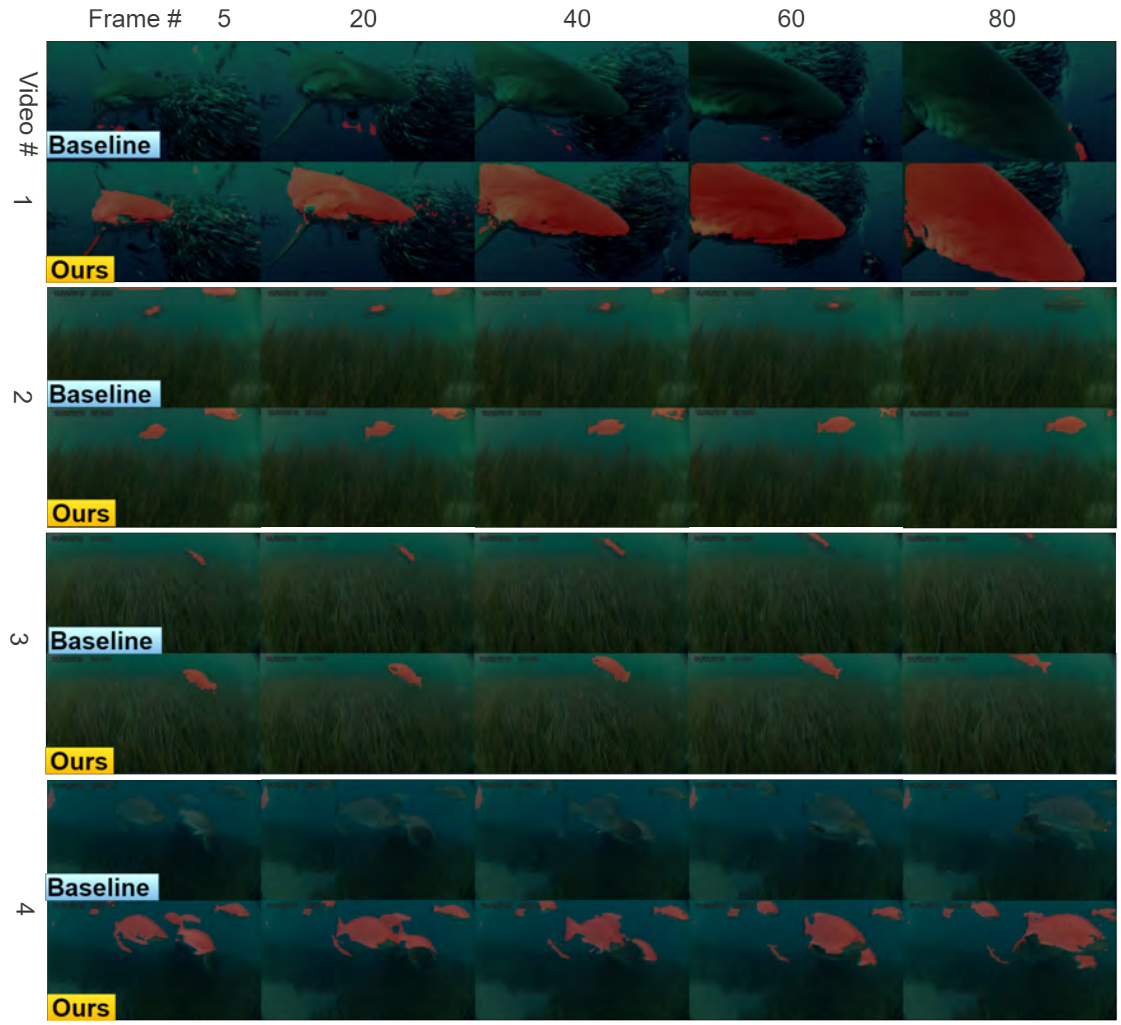

Figure 4.6: Qualitative comparison between our model and a CNN-based encoder baseline [12] model applied on the YouTube-VOS (rows 1 and 4) [16], and Seagrass (rows 2 and 3) [10] datasets. The representation learned by our model effectively distinguishes between objects and background ambiguity and is robust to occlusions.

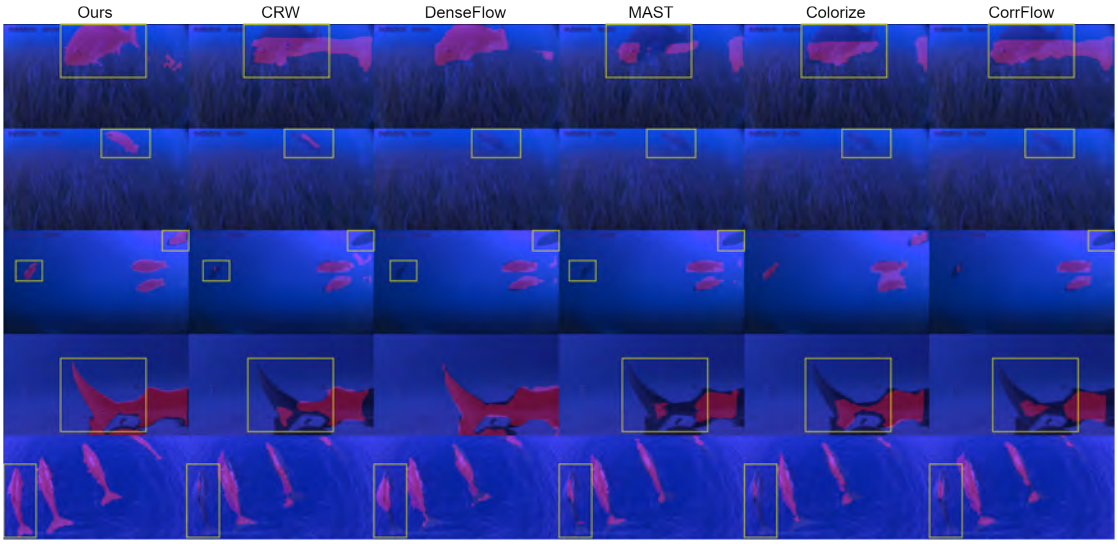

Figure 4.7: Qualitative comparison between our model and five state-of-the-art models (CRW [11], DenseFlow [12], MAST [13], Colorize [14], CorrFlow [15]) applied on Seagrass [10] (rows 1-3), and the YouTube-VOS [16] (rows 4 and 5) datasets. The yellow rectangle highlights instances where the other methods did not correctly identify the fish body or a significant part of it.

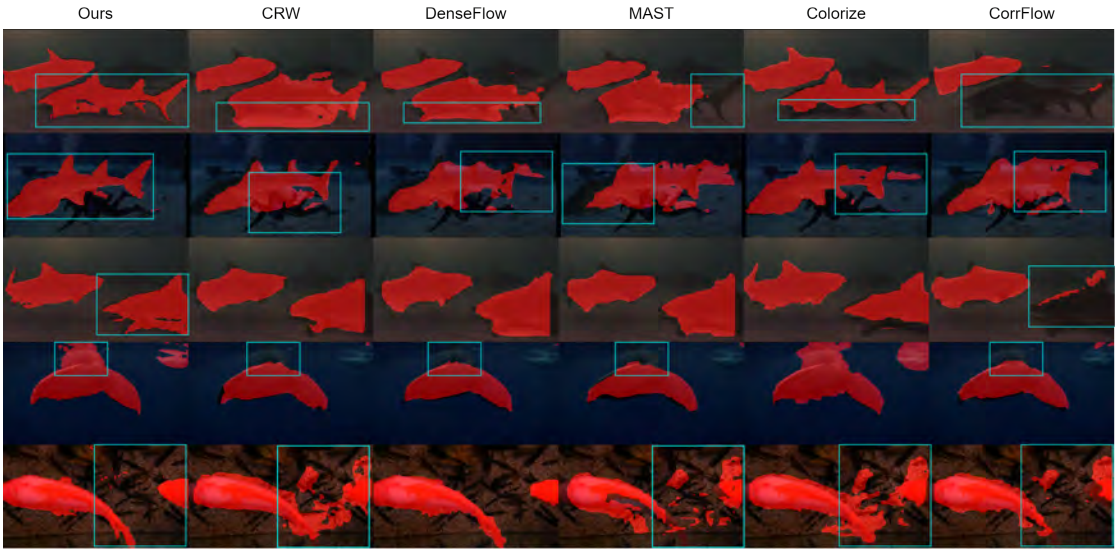

Figure 4.8: Qualitative comparison between our model and five state-of-the-art models (CRW [11], DenseFlow [12], MAST [13], Colorize [14], CorrFlow [15]) applied on the YouTube-VOS [16] dataset. The blue rectangle highlights instances where the other methods did not accurately identify the contour of the fish's body.

Table 4.3: Ablation study for other models on Seagrass [10] and YouTube-VOS [16] datasets.

	$\mathcal{J}\&\mathcal{F}_{\text{mean}}$	
Method	Seagrass [10]	YouTube-VOS [16]
Baseline [12]	45.5	60.2
FFC [309]	46.2	60.6
Segmenter [310]	41.5	52.7
PoolFormer [311]	42.8	54.9
XCiT [312]	43.7	56.8
Transformer (ours)	**50.0**	**63.3**

of the anchors could shed light on the sensitivity of our model to this hyper-parameter. We believe that these additional ablation studies could provide valuable insights into the behaviour of our model and further strengthen our findings.

4.4.9 Failure Case

In our experiments, we noticed that in a few cases, part of the background is mistakenly segmented as fish. For two examples, see Fig. 4.9. We envision that the attention model could potentially be enhanced by incorporating additional modalities to improve performance on tasks where the input contains objects that are similar to the background. These extra data modalities could include depth or motion information. This can be particularly useful in cases where the object and the background are highly similar in terms of colour or texture. We also find that when there is heavy occlusion from seagrass, the model cannot segment the whole fish's body correctly. Some instances of this problem are shown in Fig. 4.10.

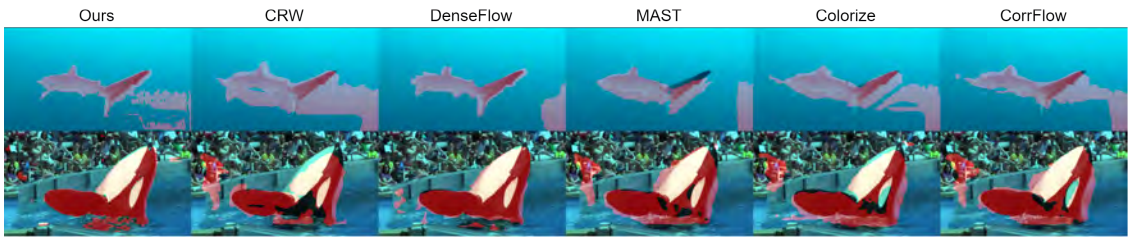

Figure 4.9: Failure case of our model applied to one frame from the YouTube-VOS [16] dataset. Our model similar to all other studied models failed to differentiate between the fish and the background.

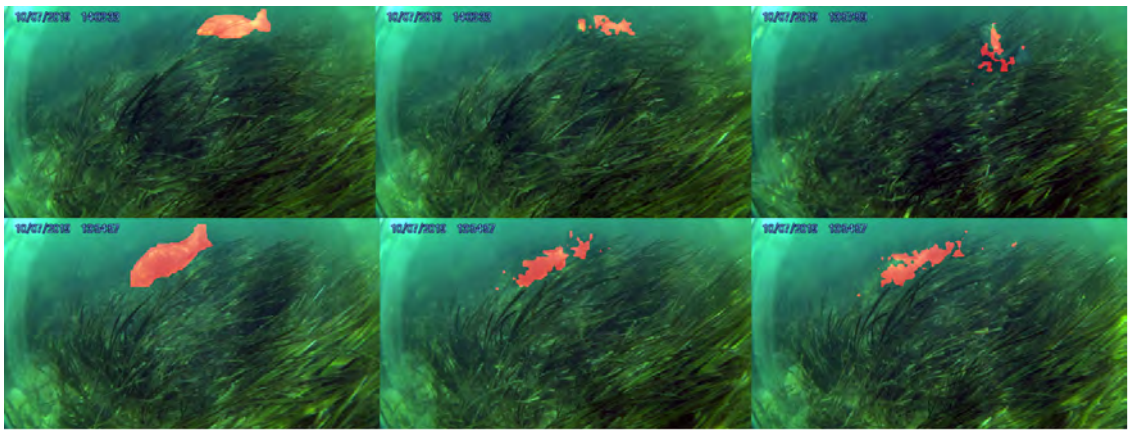

Figure 4.10: Failure case of our model applied on a sequence of video frames from the Seagrass [10] dataset. Our model failed due to heavy occlusion from seagrass.

4.5 Discussion

Underwater video object segmentation is challenging due to low visibility, variable lighting, refraction and reflection, dynamic backgrounds, and deformation of objects caused by the refraction of light through water. These factors can make it difficult to accurately segment objects in underwater video. The main aim of this study was to propose an end-to-end self-supervised deep learning method for underwater fish segmentation that addresses all these challenges while significantly improving the state-of-the-art techniques using innovative approaches. Here, we describe some of the general and specific challenges we faced when designing our model and explain how innovative solutions were used to resolve these challenges to develop a strong model.

4.5.1 Model Innovations and Strengths

We aimed to develop an approach that is robust to different underwater scenes. Previous methods have used fully-supervised learning methods, which rely on extensive annotation and are hard to apply to a wide range of underwater video applications. In contrast, our proposed method utilizes self-supervised learning to avoid the problem of reliance on large datasets and manual labelling requirements.

In addition, our model innovatively addressed the degenerate solutions, also known as overfitting or overgeneralization, which is a common concern in deep learning and can be especially important in the context of object segmentation. Degenerate solutions can occur when a model is overly complex or has too many parameters, which can lead to poor

generalization performance on new data. In our initial experiments, we found that without using a regularization branch (Fig. 4.2 second branch), the model tended to quickly converge on a trivial solution, which has also been observed by other researchers [11]. Our analysis suggested that the network was encoding positional cues into a degenerate feature representation. We believe that this may be due to the limited spatial invariance of current CNN implementations and the use of padding, which provides a predictable and stable pattern. To prevent the model from finding such shortcut solutions, our innovative solution was to replace the CNN with a Transformer, which provides a self-attention mechanism. The self-attention mechanism allows transformers to attend to different parts of an input sequence and weigh their contributions to the final output. Another innovation in our model was to use spatial jittering when sampling anchors instead of a fixed offset. These two novel modifications prevented our model from converging on degenerate solutions and improved generalization performance on new data, which is a significant advantage of our technique compared to prior video segmentation models.

Our anchor sampling technique was also novel. Anchor sampling is used for selecting a set of anchor points or regions within an image or video that are used as reference points for object detection or segmentation tasks. The choice of anchor points or regions can have a significant impact on the performance of the model, as they serve as the basis for the model's predictions. We derived all anchors from a single frame, known as the reference frame. Our method also used pseudo labels and a self-training loss function (described in Sec. 4.3.4) to align the representations of the other frames with the reference frame, which originated from the same video. This ensured that our model is able to accurately segment objects in the underwater videos despite the challenging environment.

Furthermore, inspired by class-specific representations in other contexts [313], another novel aspect of our model is computing the affinity between features and anchors extracted from multiple videos in the training batch using the equation provided in Sec. 4.3.3. This method only selects dominant anchors for self-training from the same video sequence as the feature, which means that the features will be attracted to anchors only from the same video. The distance between anchors and features from different video sequences will also increase due to the contrastive formulation of the affinity. This design allows the method to implement inter-video discrimination, which makes our model stronger.

Computational efficiency achieved through our innovative use of a grid sampling approach is another innovation in our model. Using griding to extract anchors can lead to significant computational and memory savings. For example, using a grid of size 8×8 with an image size of 32×32 pixels will result in a computational and memory-saving

factor of 16. This is particularly valuable when considering the storage costs of the affinities u, which have a size of $B \cdot T \cdot h \cdot w \times B \cdot N^2$, with respect to B and T. Also, we compute feature affinities in parallel rather than sequentially. This allows us to train our model on a single 11GB GPU, and achieve real-time inference, which is crucial for online video processing applications.

We also utilised other novel approaches for our model development to make it more practical. First, the order of the frames in a sequence within a training batch is not important and can be randomly selected, because our method does not rely on assumptions about motion continuity. Additionally, the method does not use any momentum network or queue buffers, which can make the training process more stable. In terms of data augmentation, the method only uses similarity transformations and does not require appearance-based augmentations such as photometric noise. Instead, it relies on learning natural changes in appearance directly from video data. Overall, these practical considerations make the proposed method a simple and stable solution for object segmentation in underwater videos.

4.6 Conclusion and Future Work

In conclusion, the self-supervised learning method proposed in this study has demonstrated strong performance in underwater fish segmentation, outperforming previous models and achieving results comparable to fully-supervised methods.

The proposed method has several strengths that can benefit the field of underwater video analysis. It is able to effectively segment fish in real-world underwater videos, while not requiring a pre-trained model. It is also computationally efficient, making it suitable for edge devices. Additionally, the use of self-supervised learning allows for the model to be trained without the need for extensive manual annotation, which can be time-consuming and prone to overfitting.

One way that others in the field can benefit from this work is by using it as a starting point for developing self-supervised learning approaches for other underwater video tasks, such as object tracking and counting. Also, our proposed model can be used as a generic tool for fish habitat monitoring, which is essential in marine ecology.

Despite its advantages, the main limitation of our model is that it only focuses on underwater fish segmentation, and cannot identify fish species. Another limitation of our study is the scarcity of its training data. We rely on existing publicly available training data and a relatively small number of test videos to evaluate our model. However, we

hope that this work inspires further research on larger datasets.

Future research could address some of the limitations of our study. One potential direction is to extend our model not only to segment but to also identify fish species. This could be achieved by collecting and annotating a larger dataset with multiple fish species, or by incorporating additional features such as size, which can better differentiate between different species. Another future research direction is to extend our model to other underwater objects such as corals, plants, or other marine animals. This needs collecting additional training data and labelling and adapting the model architecture or loss function to handle the different characteristics of these objects. Yet another potential direction is to explore the use of additional modalities, such as depth data or multispectral imagery, to improve the segmentation results. These additional modalities could be included in the model architecture or loss function to evaluate their impact on the segmentation performance.

Overall, this work has the potential to contribute to the field of underwater video analysis and assist in tasks such as marine and aquaculture farm monitoring, species distribution, stock management, and fishing enforcement.

Chapter 5

How to Track and Segment Fish without Human Annotations: A Self-Supervised Deep Learning Approach

Tracking fish movements and sizes is crucial for understanding their ecology and behaviour. Knowing where fish migrate, how they interact with their environment, and how their size affects their behaviour can help ecologists develop more effective conservation and management strategies to protect fish populations and their habitats. Deep learning is a promising tool to analyze fish ecology from underwater videos. However, training deep neural networks (DNNs) for fish tracking and segmentation requires high-quality labels, which are expensive to obtain. In this Chapter, the first research question is also addressed. Specifically, we propose an alternative unsupervised approach that relies on spatial and temporal variations in video data to generate noisy pseudo-ground-truth labels. We train a multi-task DNN using these pseudo-labels. Our framework consists of three stages: (1) an optical flow model generates the pseudo labels using spatial and temporal consistency between frames, (2) a self-supervised model refines the pseudo-labels incrementally, and (3) a segmentation network uses the refined labels for training. Consequently, we perform extensive experiments to validate our method on three public underwater video datasets and demonstrate its effectiveness for video annotation and segmentation. We also evaluate its robustness to different imaging conditions and discuss its limitations.

5.1 Introduction

Automatic tracking and segmentation of individual fish have a wide variety of applications in ecological behavioural analysis [1, 26, 314–316]. Understanding and predicting animal motion in the wild would bring significant benefits in many research and industry domains [317–320]. However, the movement and motion of animals in their natural environments is highly complex. Multiple factors can contribute to the complexity of individual movements. The animals are not always visible in a video, which makes tracking and segmentation difficult. Multiple animals may be in the same video, complicating the segmentation task. These challenges often require advanced computational methods.

Several studies have tried to address these challenges [9, 24, 42, 198, 321]. Such studies rely heavily on pixel-level annotations to train or improve their Deep Neural Network (DNN). These annotations are expensive and time-consuming, especially for fish segmentation in the wild. The key underlying assumption of most of the current automated methods [24, 151, 170, 194] is that the training data is usually paired with the ground truth that comes from a video that contains a large number of fish. Although ground truth is still expensive, obtaining a large number of video sequences is necessary, because achieving accurate results using a small number of sample videos is very difficult.

This study was motivated by the importance of the challenges faced when trying to annotate and segment animals in videos in the wild. Unlike in controlled conditions, where animals are easily distinguishable from the background, fish are difficult to distinguish in realistic videos [10, 17], even with domain knowledge. This is due to large variations in animal appearance, lighting conditions, and background.

Our approach aims to develop an unsupervised method for fish tracking and segmentation without the need for human annotations, by leveraging spatial and temporal variations in video data using known techniques of background subtraction and optical flow, as shown in Fig. 5.1. Specifically, we propose to generate pseudo labels based on unlabelled video data. The use of pseudo labels can benefit various learning-based algorithms since it can significantly reduce the labelling cost. The key to the proposed method is to leverage the intrinsic temporal consistency between consecutive frames to improve the generated labels by refining them with a self-supervised model. We propose to train a DNN to segment individual fish based on the generated pseudo-labels. As long as the pseudo-labels are generated in a way that they have similar structure and appearance to real ones, the model can learn to understand the underlying structure from the pseudo-labels. In general, the more realistic the pseudo-labels, the better the segmentation accuracy. We

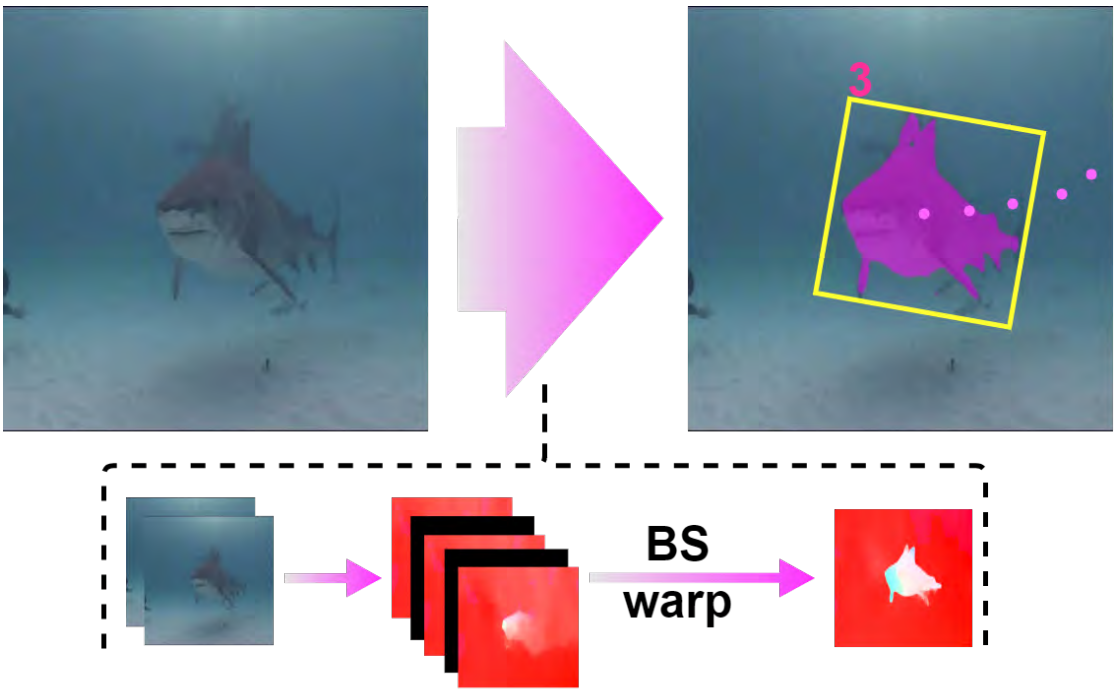

Figure 5.1: Combining background subtraction and optical flow demonstrate how both levels work in concert to preserve object boundaries and temporal coherence throughout the video. Please refer to Sec. 5.3 for details.

include a short video of our model's prediction at https://youtu.be/Z5G7YBoL3eM and https://youtu.be/8LOKsVSiY9U.

The main contributions of this paper are listed as follows:

- Propose to use pixel-level pseudo labels generated by an optical-flow model and background subtraction to learn the segmentation and tracking of individual fish automatically without manual interaction.
- Demonstrate that using self-supervised refinement, we can further improve the accuracy of the pseudo labels for fish tracking and segmentation.
- Evaluate our method on three public datasets with different image quality.
- Discuss the limitations of the current model and our future research directions.

The rest of the paper is organized as follows. Sec. 5.2 covers related works and provides background information on the novel aspects of our work. Our model's framework is described in detail in Sec. 5.3. Sec. 5.4 presents our method for training and evaluating

our model. The experimental setup and results are presented in Sec. 5.5, while detailed discussions of our results are presented in Sec. 5.6. Finally, Sec. 5.7 concludes our paper.

5.2 Related Work

In this section, we briefly review the research domains most relevant to our work.

Video object segmentation is a task that is used to locate and segment each target object [322–324]. The target object to be segmented can either be a class of interest in the videos or the moving objects of interest. Object segmentation is generally categorized into two categories: segmentation with instance-level semantics and segmentation without instance-level semantics, which is the main objective of this paper. Therefore, this study focuses on generating labels without human intervention. Some segmentation methods for moving objects have been developed by using background subtraction techniques [325–327]. Several of these approaches are based on the assumption that the scene is locally constant [328,329]. This means that the background in one frame is assumed to be similar to the background in the next frame or only a few pixels away. In order to use this assumption, they estimate the local background and threshold it according to the similarity threshold to identify foreground regions. However, this method is known to be sensitive to illumination changes, and may even lose all detail within the image due to over-estimating the local background. Another approach to segmentation uses the detection of optical flow to define motion boundaries [330–333].

Optical flow predicts the relative motion of objects in two consecutive frames of a video [332, 333]. It gives a dense correspondence between frames, but at the cost of being limited to rigid objects, and computation entangled. Additionally, optical flow can only work within scenes where the movement of the camera is significantly lower than the movement of the object [330, 331, 334]. This can be seen as a limitation, as the background subtraction method can be used in a wider range of applications. However, the key element of optical flow is that it can also be used for background subtraction [335,336]. By tracking the movement of the pixels between frames, we can determine the background. If a pixel that is part of the background does not match the static background within a given threshold, then that pixel is determined to be an instance of an object.

Another segmentation approach is based on the detection of visual motion. It is based on the fact that moving objects in the scene induce consistent changes to the flow of pixels in a region [331, 334]. However, due to substantial displacements or occlusions, their calculated optical flow may contain considerable inaccuracies [337–339]. In our method,

we address these issues and enhance both estimated optical flow and object segmentation, simultaneously.

Video object tracking is the task of assigning a consistent label for each individual object in the scene as it moves [340, 341]. This tracking is generally divided into multiple steps, including detection of the object of interest, tracking of the moving object in the scene, and then associating labels between frames. The tracking task, therefore, consists of identifying the bounding box of the object over several video frames and, at the same time, updating the location of the object in the image [342, 343]. This can be done based on a similarity metric between different frames [344, 345]. The idea of such a metric is to find the closest objects in the frame with an overlapping bounding box. This can be performed at either the pixel level or at the region level. The major drawback of this method is the computation time [346, 347] that is needed to compute all the similarities between all the different frames. On the other hand, if the computational resources are available, this method has been proven to be useful when tracking fast-moving objects and when the objects are not occluded in the frame [348, 349]. In our method, we produce the rotating 2D object bounding box from each instance mask of the object over several video frames.

Supervised And Unsupervised Learning. Supervised learning has been used to build object detection [350–352], video object segmentation [323, 324] and video object tracking [341, 342]. These methods require extensive human annotation and therefore are not suitable for video annotation in the wild. To reduce the labelling costs of data, unsupervised learning has emerged as a powerful technique for the learning of video data. In the traditional image domain, unsupervised methods are expected to outperform their supervised counterparts [24, 198, 321, 353, 354] due to their potential to train data without labels.

The idea behind many of the unsupervised DNN models is to learn a feature representation from unlabelled data [355–357]. Then, a DNN model can be applied to the learned feature representation to produce the output. For example, in the domain of video segmentation [9, 125, 288, 289], DNNs have been used to learn a representation from the difference between a pair of unlabelled videos [11, 12, 303] and from warped frames [358].

In this work, we focus on unsupervised learning. Our proposed method will generate labels referred to as pseudo-labels to train a multi-task supervised DNN for video object segmentation and video object tracking.

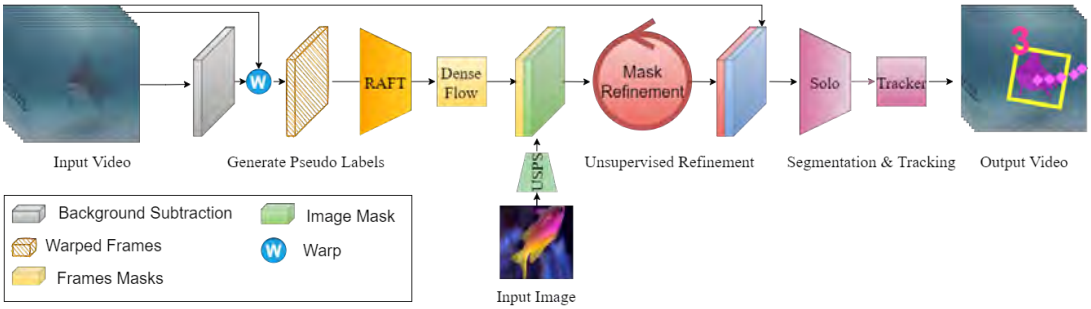

Figure 5.2: Our proposed framework consists of three main components: generate pseudo-labels, unsupervised pseudo-labels refinement, and segmentation network. The proposed segmentation model trains with the generated pseudo-labels, which are refined with self-supervised training. Please refer to Sec. 5.3 for details.

5.3 Framework

The overall framework can be divided into three stages as shown in Fig. 5.2. The first stage is to generate pseudo-labels using background subtraction and optical flow for both videos and still images. The second stage is to train a self-supervised model to refine the pseudo-labels using their spatial structure. In the last stage, the refinement of the video and still-image versions are applied jointly to train the segmentation network and to predict the final label. The segmentation network's training behavior closely matches supervised training because we employ improved pseudo-labels. As a result, the network's training process is more reliable than that of current unsupervised learning techniques [3, 11, 12, 303]. In the following subsections, we describe the details of these three components and the corresponding loss functions.

5.3.1 Background Subtraction

As a first step to generating pseudo labels, background subtraction is performed on the video frames. A clean background image is estimated for every video sequence by computing the median of the first 10 frames of the video sequence along the first axis. This is to average out any distracting elements that come in front of the clean background. Then, each video frame is subtracted from the clean background to create the mask sequence. After the subtraction, all the foreground pixels take on a value of 1, and pixels belonging to any background region have 0 values using Adaptive Gaussian Thresholding [359].

Adaptive Gaussian Thresholding is used instead of one global value as a threshold because it sets a pixel's threshold based on a local region surrounding it. As a consequence,

we obtain various thresholds for various areas inside the same image, which produces better results for images with varying illumination.

This background subtraction step is crucial in eliminating any stationary elements or shadows from the video sequences that might disturb the next step, optical flow.

5.3.2 Optical Flow

The next step of pseudo-label generation is to compute the optical flow using Recurrent All-Pairs Field Transforms (RAFT) [360]. However, optical flow is frequently inaccurate at object boundaries, in which we want our segmentation to be accurate exactly at these borders. Therefore, we consider video segmentation from background subtraction and optical flow estimation simultaneously. Using pixel level and temporal information sources, the segmentation algorithm is improved by removing the artifacts induced by background subtraction and optical flow. We demonstrate how both levels work in concert to preserve object boundaries and temporal coherence throughout the video. The key is that we need to remove motion blurs while preserving the motion of the fish boundaries.

To achieve the pseudo labels, we first deconstruct a pair of video frames, x_t and x_{t+1}, and estimate a mask m_t and m_{t+1} with the background subtraction method as described in section Sec. 5.3.1. The segmented masks m_t, m_{t+1} are used to synthesize frames $\hat{x_t}$ and $\hat{x_{t+1}}$ by warping x_t and x_{t+1} with m_t, m_{t+1}, respectively. The optical flow [360] takes two frames $\hat{x_t}$ and $\hat{x_{t+1}}$, and produces a motion vector $\hat{v}$ between them. This motion vector is used to compute the magnitude and angle of the motion. Specifically, the pixels with a motion vector $\hat{v}$ outside m_t (and m_{t+1}) are assigned the value of the background and the pixels with a motion vector $\hat{v}$ inside m_t (and m_{t+1}) are reassigned the object. We denote the reassigned images as $\hat{x_t^*}$ and $\hat{x_{t+1}^*}$ and use them as input for our segmentation step, as shown in the top panel of Fig. 5.2.

We show the optical flow results for the three video datasets with and without background subtraction of frames x_t and x_{t+1} in Fig. 5.4, Fig. 5.5, and Fig. 5.6. A mask m_{t+1} that better distinguishes the background from the foreground from the optical flow step is then refined with our proposed unsupervised refinement method in the next section. A sample optical flow comparison video before and after background subtraction is available at https://youtu.be/8LOKsVSiY9U.

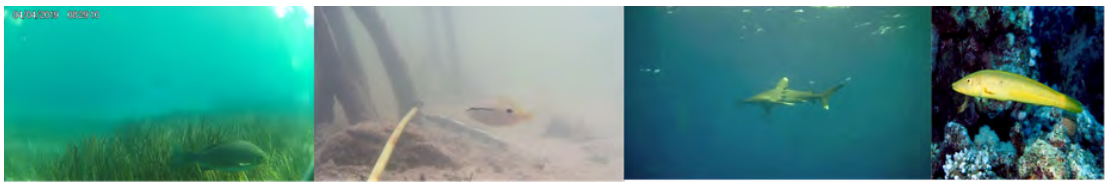

Figure 5.3: Sample image from each of the four utilised datasets. From left: Seagrass [10], DeepFish [17], YouTube-VOS [16], and Mediterranean Fish Species [34]

5.3.3 Unsupervised Refinement

The second stage in our method is cumulative pseudo-label refining via unsupervised historical moving averages (MVA) [361] using DeepLabv3 [362] network for semantic segmentation and Conditional Random Fields (CRF) [363] by minimising the F-score until the MVA predictions reach a stable state. The CRF can "sharpen" initial location predictions to make them more accurate and consistent with edges and parts of the source image that have a constant colour.

Given the pseudo-labels of the previous step, we train the network for 50 epochs. The number of epochs is low to avoid a significant over-fitting of the network to the noisy pseudo-labels. Then, the network is reinitialized with trained weight to predict a new set of pseudo-labels to train on again.

Let D be the set of training examples and M be the network model. By $M(x, p)$ we denote the mask prediction of model M over pixel p of image $x \in D$. During this stage, a historical moving average (MVA) from the last training stage is composed as follows:

$$\text{MVA}(x, p, k) = (1 - \alpha) * CRF(M(x, p)) + \alpha * \text{MVA}(x, p, k - 1),$$

where $M(x, p)$ is the network mask prediction, k is the epoch number, α is a positive real factor, and CRF is the Conditional Random Fields (CRF) [363].

We use $L_\beta = 1 - F_\beta$ as an *image-level loss* function w.r.t. each training example x. F-score (F_β) is the harmonic mean of precision and recall of the prediction output of pixel p over image x w.r.t. the pseudo-labels, that use a positive real factor β as follows:

$$F_\beta = \left(1 + \beta^2\right) \frac{\text{precision} \cdot \text{recall}}{\beta^2 \, \text{precision} + \text{recall}}.$$

The network is retrained until the MVA reaches a stable state, as shown in the middle panel of Fig. 5.2. By doing so, the quality of pseudo-labels is improved over time.

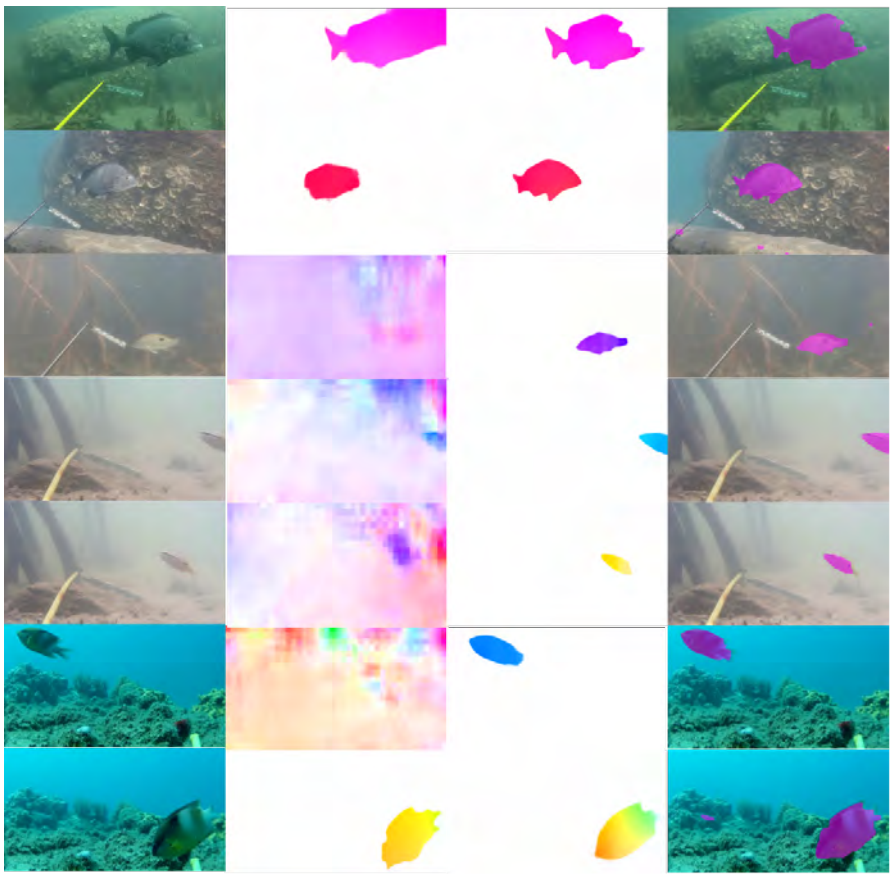

Figure 5.4: Sample optical flow results for Seagrass [10]. From left, the original image, optical flow without background subtraction, optical flow with background subtraction, mask overlay.

5.3.4 Segmenting Objects by Locations

Our last stage is training a supervised segmentation model using the refined pseudo-labels from the previous stage. The supervised model is based on Segmenting Objects by Locations (SoloV2) [364]. SoloV2 is an updated version of Solo [365], a previous method for instance segmentation. The idea is to dynamically segment objects by location.

Given an image as input, the network generates the object mask, then the object mask generation is decoupled into a mask kernel prediction and mask feature learning. Furthermore, matrix non-maximum suppression (MNMS) is applied to reduce inference overhead. Specifically, SoloV2 is composed of two modules: (1) Dynamic Instance Segmentation and (2) Matrix non-maximum suppression (MNMS). The dynamic instance segmentation scheme dynamically segments objects by locations by learning the mask kernels and mask features separately. The mask kernels are predicted dynamically by

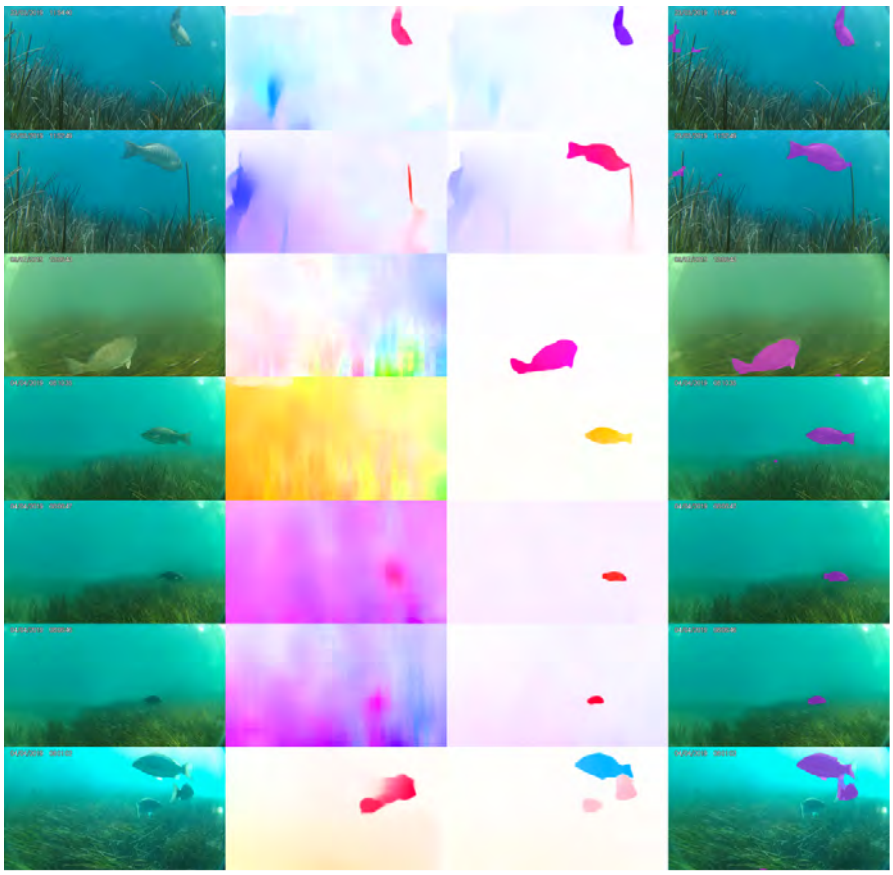

Figure 5.5: Sample optical flow results for DeepFish [17]. From left, the original image, optical flow without background subtraction, optical flow with background subtraction, mask overlay.

the FCN [232] when classifying the pixels into different location categories, then constructing a unified mask feature representation for instance-aware segmentation. The non-maximum suppression process is achieved by performing NMS with a parallel matrix operation in one shot to reduce inference overhead and suppress duplicate predictions. Compared to the widely adopted multi-class NMS [366], where the sequential and recursive operations result in non-negligible latency, the parallel non-maximum suppression with matrix operation can achieve similar performance with much lower latency. The parallel processing strategy performs the inference for the MNMS on-the-fly and enables processing at a high frame rate (34 *frames per second*). For more details, we refer the readers to [364].

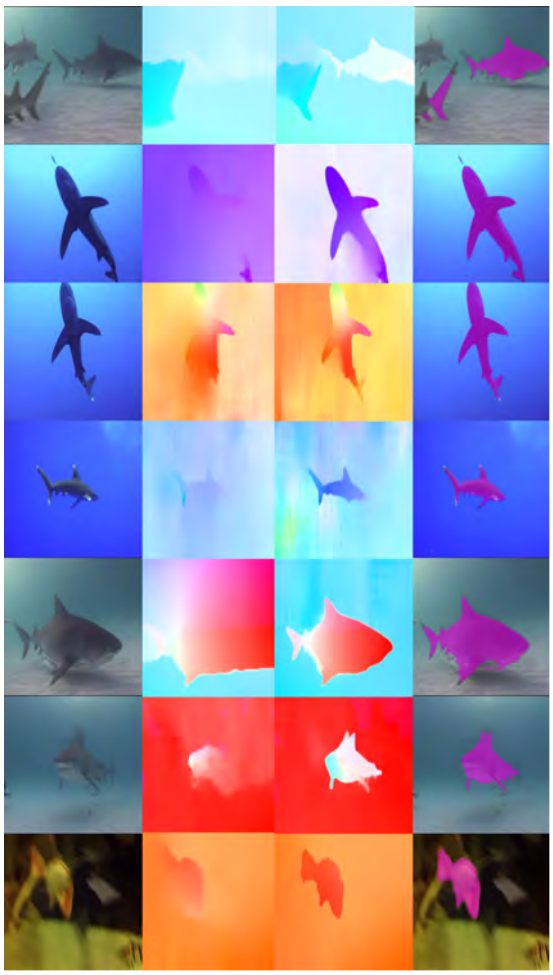

Figure 5.6: Sample optical flow results for YouTube-VOS [16]. From left, the original image, optical flow without background subtraction, optical flow with background subtraction, mask overlay.

5.3.5 Rotating Bounding-box

From each instance mask that we predicted from the previous stage, we are able to produce the rotating 2D object bounding box. The minimum bounding rectangle (MBR) technique is used to obtain a rotated bounding box from a binary mask of the object. We used OpenCV [367] to find the minimum area of a rotated rectangle. It takes the binary mask of the object as an input and returns a Box2D structure that contains the following information: (centre (x, y), (width, height), angle of rotation). The output of this step is used to track the objects as discussed in the following section.

Figure 5.7: Sample fish trajectory results. Zoom-in for better view. See also a short video of fish trajectory results at https://youtu.be/Z5G7YBoL3eM.

5.3.6 Online Tracking

We used Simple Online and Real-Time Tracking (SORT) [368] as an online tracking framework that focuses on frame-to-frame prediction and association. The bounding box position and size are only used for both motion estimation and data association. Kalman filter [369] is used to handle the motion estimation and the Hungarian method [370] is used for data association.

Motion estimation is used to propagate a targets identity into the next frame. The inter-frame displacements of each object are approximated with a linear constant velocity estimation. The detected bounding box is used to update the target state where the Kalman filter [369] solves the velocity components. The state of each target is estimated as:

$$x = [h, v, s, r, \hat{h}, \hat{v}, \hat{s}]^T,$$

where h and v represent the horizontal and vertical pixel location of the center of the target, while s and r represent the scale and the aspect ratio of the targets bounding box, respectively. Here, $\hat{h}, \hat{v}, \hat{s}$ are for the source.

Data association is assigning new detections to existing targets. Each targets bounding box is estimated by predicting its new location in the current frame. The intersection-over-union (IOU) distance between each detection and each forecasted bounding box from the existing targets is used to calculate the assignment cost matrix. The assignment cost matrix is then resolved using the Hungarian technique [370] to produce the fish trajectory as shown in Fig. 5.7.

5.4 Method

This section describes our method in detail. Our method is based on three main components: the pseudo labels generation, the unsupervised learning method to refine the generated pseudo labels and the DNN for fish tracking and segmentation. Fig. 5.2 shows the algorithm flow diagram for the fish tracking and segmentation framework.

5.4.1 Datasets

We performed experiments using four publicly available datasets, i.e. Seagrass [10], DeepFish [17], YouTube-VOS [16], and Mediterranean Fish Species [34]. Fig. 5.3 demonstrates a sample image from each dataset.

Seagrass [10] is comprised of annotated footage of *Girella tricuspidata* in two estuary systems in south-east Queensland, Australia. The raw data was obtained using submerged action cameras (HD $1080p$). The dataset includes 4280 video frames and 9429 annotations. Each annotation includes segmentation masks that outline the species as a polygon.

DeepFish [17] consists of a large number of videos collected from 20 different habitats in remote coastal marine environments of tropical Australia. The video clips were captured in full HD resolution (1920×1080 pixels) using a digital camera. In total, the number of video frames taken is about $40k$.

YouTube-VOS [16] is a video object segmentation dataset that contains 4453 YouTube video clips and 94 object categories. The videos have pixel-level ground truth annotations for every $5th$ frame ($6fps$). For a fair comparison, we extracted only the videos that contained *fish*, which include 130 video clips and 4349 video frames in total.

Mediterranean Fish Species [34] consists of a large number of images collected from 20 different Mediterranean fish species. In total, the number of images is about $40k$. The dataset was split into two subfolders, training and test sets. The training set contains $34k$ and the test set contains $6k$ images. The image resolution ranges between (220×210 pixels) and (1920×1080 pixels). The original images are stored in an RGB file format in subfolders as a class label.

We train our feature extractor on all of the four datasets and evaluate it on the video datasets only, Seagrass [10], DeepFish [17], and YouTube-VOS [16].

5.4.2 Pseudo Labeling

To train our supervised model, which was explained in Sec. 5.3.4, we first generate pseudo labels for the image dataset, Mediterranean Fish Species [34] and the video datasets, Seagrass [10], DeepFish [17], and YouTube-VOS [16].

Image Dataset

Since our image dataset [34] is curated from static images of different fish species, our framework discussed in Sec. 5.3 was not applicable to this dataset. Therefore, we used DeepUSPS [361] as an unsupervised saliency prediction network for a pseudo-labels generation. DeepUSPS is trained on the unlabelled MSRA-B dataset [371] for predicting salient objects. And it is an unsupervised learning method that produces pseudo labels with high intra-class variations, which is useful for the training of the supervised model.

However, DeepUSPS is only good in pseudo prediction for a single object in the image that is not disturbed by additional intricate details, which is not ideal for the more challenging video datasets [10, 16, 17].

Video Datasets

Unlike our image dataset, our video datasets contain both multiple objects in a single frame as well as across multiple frames. Therefore, we adapted our pseudo-label generation framework discussed in Sec. 5.3 that is capable of predicting multiple salient objects in the same video clip and handling the case of a cluttered background. This pseudo-label generation framework aims to tackle the issue of single-image datasets by generating more pseudo labels with intra-class variations in image space.

The Pseudo-label generation framework consists of three steps:

1. Obtain salient objects by performing background subtraction using Adaptive Gaussian Thresholding [359], as explained in Sec. 5.3.1.

2. Enhance the obtained salient object boundaries from the previous step with optical flow using RAFT [360], as explained in Sec. 5.3.2.

3. Apply cumulative pseudo-label refining via unsupervised historical moving averages (MVA) [361], as explained in Sec. 5.3.3.

In this way, we can get pseudo labels for video datasets, Seagrass [10], DeepFish [17], and YouTube-VOS [16], which are used to train the supervised model.

5.4.3 Model training

Our models were trained with an input resolution of 256×256 pixels. We scale the lowest side of the video frames to 256 and then extract random crops of size 256×256. We sample two video sets, $B = 2$ (of size $T = 5$ frames), therefore, $B \times T = 2 \times 5 = 10$ frames are used per forward pass.

We found that for this problem set, a learning rate of 1×10^{-3} works the best. It took around 300 epochs for all models to train on this problem. Our networks were trained on a Linux host with a single NVidia GeForce RTX 2080 Ti GPU with 11 GB of memory, using Pytorch framework [308]. We used stochastic gradient descent (SGD) optimiser [204] with an initial learning rate of 0.01, which is then divided by 10 at $27th$ and again at $33th$ epoch. We use light augmentation (resizing, grayscale). Following [364, 372], a scale jitter is used, where the shorter image side is randomly sampled from 640 to 800 pixels.

We applied the same hyperparameter configuration for all of the models. However, the optimum model configuration will depend on the application, hence, these results are not intended to represent a complete search of model configurations.

5.4.4 Inference

During tracking, we extract frames from the input video, forward each frame through the network, and obtain the fish category score from the classification branch. Initially, to filter out predictions with low confidence, we use a threshold of 0.1 and perform convolution on the mask feature using corresponding predicted mask kernels. Then, after having applied a per-pixel sigmoid, we binarise the output of the mask branch at the threshold of 0.5. The final step is the Matrix NMS, which fits the output mask with the Min-max box.

Our model operates online without any adaptation to the video sequence. On a single NVidia GeForce RTX 2080 Ti GPU, we measured an average speed of 34 frames per second.

Figure 5.8: Sample images from our model results for DeepFish [17], From left, the original image, the ground truth, the predicted image.

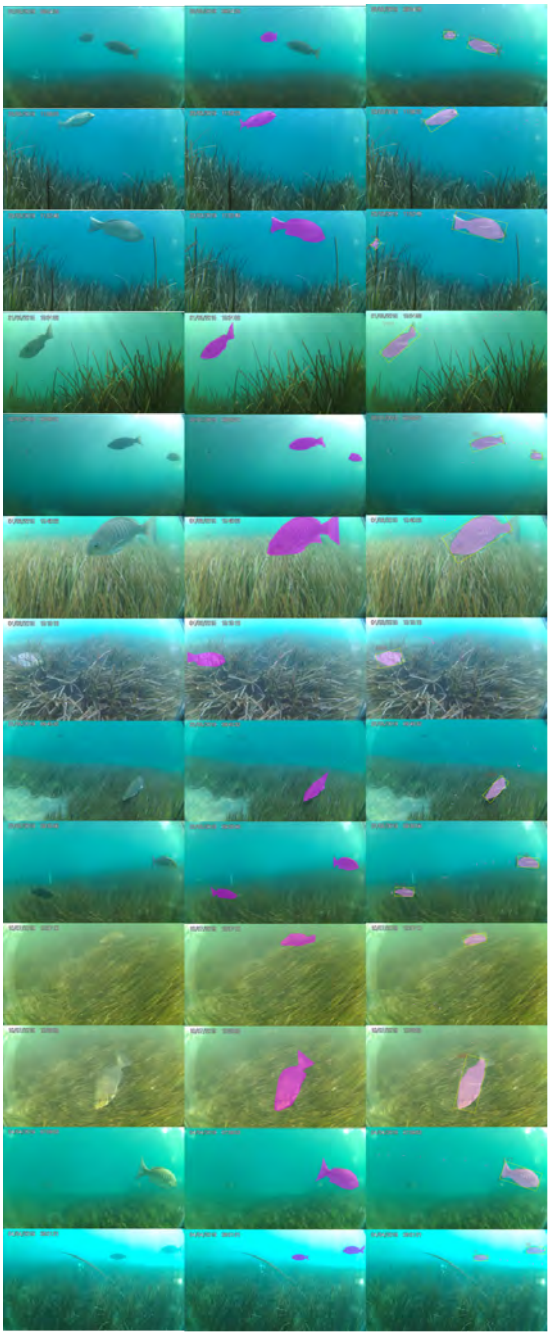

Figure 5.9: Sample images from our model results for Seagrass [10], From left, the original image, the ground truth, the predicted image.

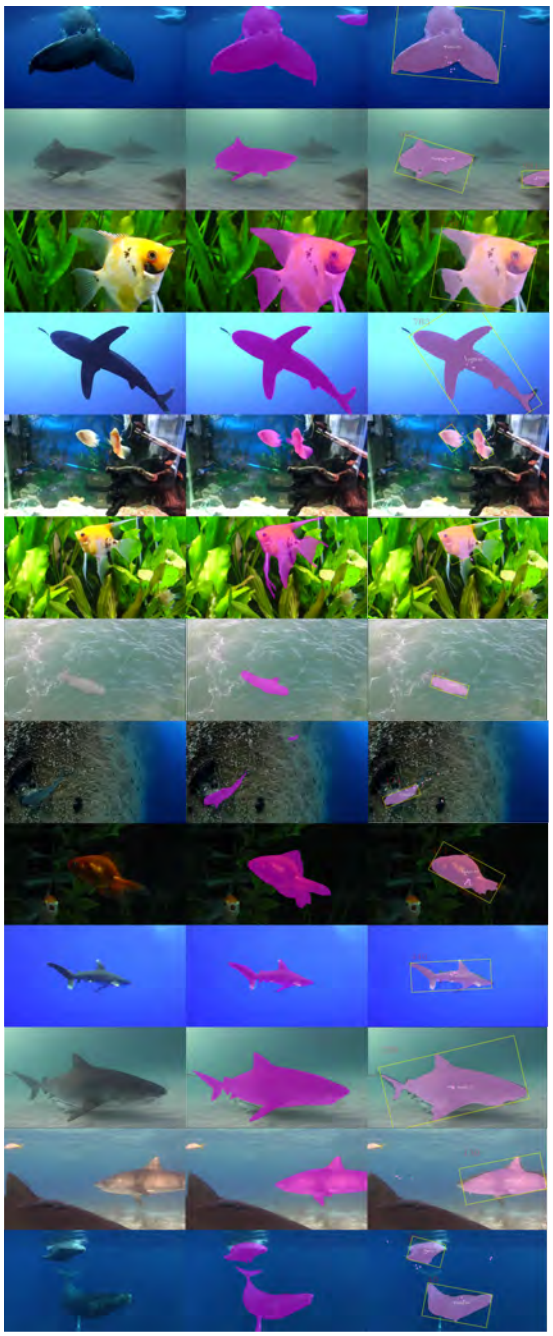

Figure 5.10: Sample images from our model results for YouTube-VOS [16], From left, the original image, the ground truth, the predicted image.

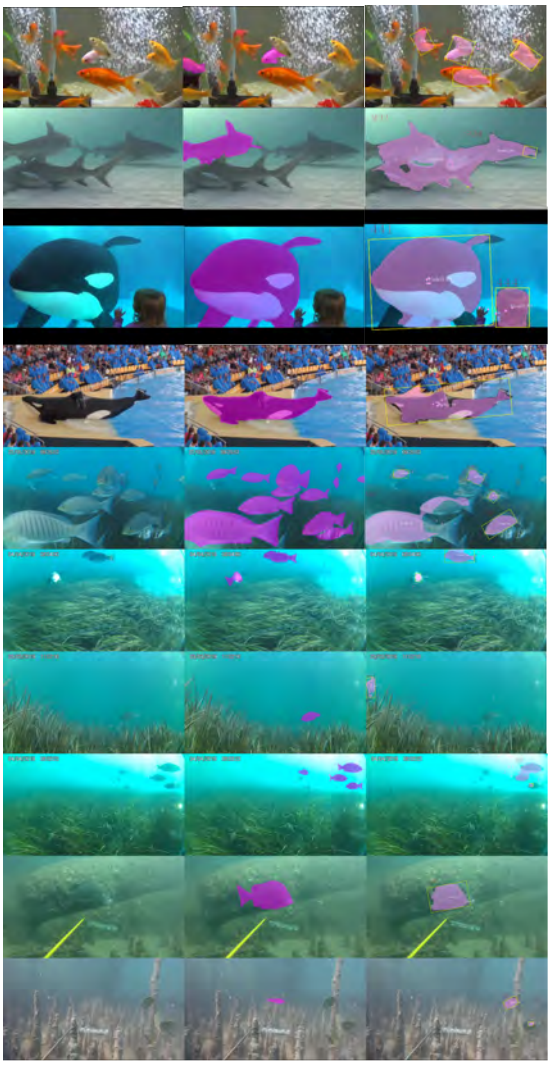

Figure 5.11: Sample images for failure cases, From left, the original image, the ground truth, the predicted image.

5.5 Experiments

We report experimental results for our model's trained representation on 50% of the DeepFish, Seagrass, YouTube-VOS datasets and the train set of the Mediterranean Fish Species dataset. We then evaluate it on the other 50% of the first three datasets. We provide quantitative and qualitative results that demonstrate our model's generalization capabilities to a range of different underwater habitats.

5.5.1 Results

We summarize our main results on Seagrass [10], DeepFish [17] and YouTube-VOS [16] datasets in Table 5.1. The quantitative results for all datasets were obtained using the COCO dataset [264] evaluation script. The average precision (AP), the average recall (AR), and Intersection over Union (IoU) were measured for the predicted bounding boxes and segmentation masks in the output images obtained from the trained SoloV2 [364], as explained in Sec. 5.3.4 in detail.

The $AP^{.50}$ and $AP^{.75}$ values were measured with the IoU thresholds of 0.5 and 0.75, respectively. The AP^M and AR^M values show the average of the AP and AR values for 10 IoU thresholds of $.50 : .05 : .95$, respectively. Averaging over $IoUs$ rewards detectors with better localization. The AP^L and AR^L values show the AP and AR values over a large area ($> 96^2$). This area is measured as the number of $pixels$ in the segmentation mask. We observed that the performance of the proposed model is stable across these three datasets. This validates the proposed model's ability to generalize well to unseen videos in other environments.

To the best of our knowledge, there is no previous work reporting detection and segmentation evaluation for these datasets. To compare our proposed unsupervised method to a supervised approach, SoloV2 [364] results on the three datasets are reported with the authors' implementation in Table 5.2. This Table shows the results of a fully supervised model with the original labels, not our generated pseudo-label.

The close accuracy results of our proposed unsupervised method compared to the original supervised SoloV2 [364] in both the detection and segmentation experiments, validate our generative approach. Moreover, our results indicate that the proposed model is not heavily influenced by different underwater habitats. The performance is on par for DeepFish [17] and Seagrass [10]. Seagrass [10] is relatively challenging since the fish are not as easy to visually detect as in the other datasets. In some cases, the proposed model is not as good as the fully supervised approaches. However, the main purpose of this study

is to develop an unsupervised method for fish tracking and segmentation. We believe that our proposed approach is more stable during training than other unsupervised methods that do not include a dedicated pseudo-label generation step.

Table 5.1: Comparison of ***unsupervised*** detection and segmentation on Seagrass [10], DeepFish [17] and YouTube-VOS [16] datasets.

Dataset	AP^M	$AP^{.50}$	$AP^{.75}$	AP^L	AR^M	AR^L
Evaluating Detection:						
Seagrass [10]	22.1	72.5	13.7	38.2	61.4	61.3
DeepFish [17]	11.7	35.0	05.3	19.3	34.5	57.1
YouTube-VOS [16]	23.6	43.2	18.4	26.9	46.1	57.5
Evaluating Segmentation:						
Seagrass [10]	12.0	37.6	05.2	20.8	31.2	52.0
DeepFish [17]	31.2	75.0	24.4	43.8	56.6	59.4
YouTube-VOS [16]	15.4	33.0	12.2	19.2	33.8	42.2

Qualitative results of our algorithm for DeepFish [17], Seagrass [10] and YouTube-VOS [16] datasets are shown in Fig. 5.8, Fig. 5.9 and Fig. 5.10, respectively. We also include additional examples of failure cases in Fig. 5.11. Overall, especially for non-rigid objects, the proposed algorithm produces favourable outcomes in the majority of images. This is despite the fast movements or crowded and complicated backgrounds causing these images to frequently have significant distortion. See also a short video of our model's prediction at https://youtu.be/Z5G7YBoL3eM.

5.5.2 Ablation Study

We performed an ablation study to demonstrate the proposed approach's effectiveness in generating pseudo labels. Specifically, we analyzed the contribution of the vital component in the proposed method, the optical flow with background subtraction (Sec. 5.3.2). In addition, we evaluated the segmentation network training with refined pseudo-labels (Sec. 5.3.4) for different epochs. The results reported in Table 5.3 are for the unsupervised segmentation based on optical flow without background subtraction as a baseline.

Table 5.2: Comparison of ***supervised*** detection and segmentation on Seagrass [10], DeepFish [17] and YouTube-VOS [16] datasets.

Dataset	AP^M	$AP^{.50}$	$AP^{.75}$	AP^L	AR^M	AR^L
Evaluating Detection:						
Seagrass [10]	32.4	82.2	13.0	34.9	68.5	72.4
DeepFish [17]	12.2	41.8	04.3	20.9	41.0	68.0
YouTube-VOS [16]	25.9	56.2	21.6	32.8	54.1	69.9
Evaluating Segmentation:						
Seagrass [10]	18.0	56.4	07.8	31.2	36.8	68.0
DeepFish [17]	46.8	72.5	36.6	50.7	64.9	72.1
YouTube-VOS [16]	23.1	49.5	18.3	28.8	40.7	53.3

And the results reported in Table 5.4 are for the four epochs trials with the same random seeds, please refer to Sec. 5.4.3 for the details.

It is apparent from the results that the segmentation accuracy of our proposed method has improved significantly when compared to that of the baseline method. We also note that the accuracy of the models also depends on the number of epochs used in the training. We observe from the results shown in Table 5.4 that the segmentation accuracy decreases after 100 epochs. The reason for this is the over-fitting of the network to the noisy pseudo-labels. While the training losses for both the baseline and our model decreased, the segmentation accuracy for our model was still greater than that for the baseline.

As part of our ablation study, we experimented with other different threshold values in addition to (0.1 and 0.5) values in Sec. 5.4.4. Our findings revealed that the performance of our proposed approach was not significantly impacted by the choice of threshold value.

5.6 Discussion

Fish segmentation and tracking are notoriously difficult tasks, especially for small fish in video data where the background, lighting conditions and fish shape can vary significantly. In particular, for real data, the quality of ground truth labels varies from video to video since it is difficult to annotate the animal's entire path. Therefore, our model aims to

Table 5.3: Comparison of ***unsupervised*** segmentation based on optical flow *without* background subtraction.

Dataset	AP^M	$AP^{.50}$	$AP^{.75}$	AP^L	AR^M	AR^L
Evaluating Segmentation:						
Seagrass [10]	05.0	23.8	03.1	14.7	19.7	29.5
DeepFish [17]	15.3	44.8	13.6	33.5	42.7	37.4
YouTube-VOS [16]	07.2	23.8	07.4	11.9	26.1	33.0

generate a pseudo ground truth by leveraging temporal consistency between frames and improving its quality based on self-supervised learning. The key to our proposed model is to leverage the intrinsic temporal consistency between consecutive frames by using the optical flow and background subtraction method to improve the generated labels. This is especially important when the fish is moving quickly and not in the same location in the consecutive frames, as is the case in natural data. Tracking fish in video data is also challenging because their motion is very irregular and small fish may not be visible throughout the entire dataset. The other problem is that segmentation and tracking are time-consuming tasks, especially when dealing with large datasets.

Our model outperforms the baseline method (the optical flow without background subtraction) with higher AP values in most of the cases. Our approach can utilize temporal consistency to produce consistent labels. In the case of the DeepFish dataset [17], we observed that our proposed unsupervised model results in higher accuracy compared to the Seagrass dataset [10]. This is mainly due to the more challenging videos in the Seagress dataset [10] compared to the video data in DeepFish [17]. Furthermore, we show that for different video datasets, our model shows similar accuracy. Therefore, we can expect that the accuracy would be similar when tested under the same conditions but in new underwater video datasets.

In addition, the segmentation accuracy does not degrade after training with supervised training, and training converges in only a few epochs, as shown in Table 5.4. In our experiments, we found that the segmentation quality has a significant impact on tracking performance. This is because the quality of the produced object bounding box has a high impact on tracking performance. Even in this case, we still achieved decent results.

We also analyzed the robustness of our proposed model with respect to the environmental conditions. We observed degradation of the model's performance when several

Table 5.4: Comparison of ***unsupervised*** segmentation for different epochs: 50,100,150,300

Dataset	AP^M	$AP^{.50}$	$AP^{.75}$	AP^L	AR^M	AR^L
50 epochs:						
Seagrass [10]	12.4	33.6	07.2	20.4	28.4	47.2
DeepFish [17]	32.0	68.6	30.8	34.8	53.6	56.2
YouTube-VOS [16]	15.8	34.0	13.8	19.8	33.8	42.2
100 epochs:						
Seagrass [10]	12.0	37.6	05.2	20.8	31.2	52.0
DeepFish [17]	31.2	75.0	24.4	43.8	56.6	59.4
YouTube-VOS [16]	15.4	33.0	12.2	19.2	33.8	42.2
150 epochs:						
Seagrass [10]	12.0	36.0	04.8	20.4	30.0	48.8
DeepFish [17]	30.4	69.8	23.2	32.4	54.2	56.8
YouTube-VOS [16]	15.2	34.0	14.0	20.2	32.8	41.0
300 epochs:						
Seagrass [10]	10.8	33.6	04.0	18.8	28.0	46.4
DeepFish [17]	29.8	70.0	22.4	31.8	53.0	55.6
YouTube-VOS [16]	15.2	33.8	14.4	23.0	32.0	40.0

fish were heavily occluded, like in Fig. 5.11. However, our proposed model is still able to estimate the fish mask in some parts as long as they are part of the animal body. One of the main challenges in this task is the large variability in the size and shape of fish, as well as the variation in the shape of the fish's body. While it is possible to identify a certain shape of fish, it is not always possible to determine the number of fish in the image.

Given a set of unlabeled video collections, the main limitation of our study is that it is only capable of segmenting foreground items and cannot distinguish between distinct object instances or semantic classes. Occasionally, the whole object or parts of the object

may not be segmented out. Our model's performance is highly influenced by the characteristics of training videos, the coverage of object categories, and the motion of both the camera and the objects, similar to other data-driven learning techniques. Our results are based on a few assumptions. One is that a small subset of semantically similar objects (e.g., all fish) exists in the scene, and these objects are likely to share the same motion feature or to be semantically similar. These assumptions are reasonable if the objects are within a certain size range, they all belong to the same class, and most of them share similar colours, shapes, and sizes. Another limitation of our approach is that we used a relatively large number of videos with a relatively small number of object categories (for instance compared to ImageNet). This allows our model to segment objects of all shapes and colours with only a handful of training examples.

One other limitation of our current framework is that, in some cases, it is unable to detect all the objects that appear in the video. In future work, we intend to study how to develop a detection-based model that is able to detect all the objects appearing in a given scene. Therefore, in the next step, we should look for a more robust and generic objectness model that is able to generalize across a variety of object categories, and a variety of background types. Further work could be conducted on more fine-grained object segmentation, especially with new video datasets.

Moreover, background subtraction is a process of detecting foreground from video streams and has drawn much attention due to its applications in multiple domains such as automated video surveillance, Human-Machine Interaction (HMI), content-based video coding, anomaly detection, visual analysis of human activities, visual observation of animals, and target tracking. However, the robustness of background subtraction is affected by various challenges such as illumination variations, shadows, dynamic background, camouflage, intermittent object motion, moving cameras, night videos, low frame rate, and bootstrapping. Additionally, the influence of illumination variation, change in the appearance of moving objects, and the presence of abrupt motion, occlusion and shadow can also cause the background compensation method to fail to detect moving regions resulting in moving objects escaping detection. These are some scenarios where the background subtraction method may fail to accurately detect motions.

5.7 Conclusion

We proposed a novel unsupervised method for tracking and segmenting fish in videos in the wild. Our results demonstrate that our proposed pseudo-label generation method

that combines optical flow with background subtraction followed by an unsupervised refinement network, leads to accurate segmentation results when used to train a supervised segmentation DNN. We showed that this segmentation approach can be used for effective tracking. We tested our model on three challenging datasets with the results indicating that our method could be a valuable candidate for assisting in the video processing of fish behaviour. Future work can extend our approach to other animal species that are commonly found in aquatic environments. Another important direction is to extend this model to other fields, for example, tracking-by-detection for autonomous driving.

Chapter 6

Adaptive Uncertainty Distribution in Deep Learning for Unsupervised Underwater Image Enhancement

One of the main challenges in deep learning-based underwater image enhancement is the limited availability of high-quality training data. Underwater images are difficult to capture and are often of poor quality due to the distortion and loss of colour and contrast in water. This makes it difficult to train supervised deep learning models on large and diverse datasets, which can limit the model's performance. In this Chapter, the second research question is addressed. Specifically, we explore an alternative approach to supervised underwater image enhancement. Therefore, we propose a novel framework called Uncertainty Distribution Network (*UDnet*), which learns to adapt to Uncertainty Distribution in its unsupervised reference map (label) generation to produce enhanced output images. UDnet is composed of three main parts. A raw underwater image is first adjusted for contrast, saturation, and gamma correction; one of these adjusted images is then randomly fed to (1) a statistically guided multi-colour space stretch (SGMCSS) module that generates a reference map to be used by (2) a U-Net-like conditional variational autoencoder (cVAE) module, to extract features for feeding to (3) a probabilistic adaptive instance normalization (PAdaIN) block that encodes feature uncertainties for final enhanced image generation. We use the SGMCSS module to ensure visual consistency with the raw input image and to provide an alternative to training using a ground truth image. Hence, UDnet does not need manual human annotation and can learn with a limited amount of data to achieve state-of-the-art results. We evaluated UDnet on eight publicly-available datasets. The results show that it yields competitive performance compared to other state-of-the-art approaches in quantitative as well as qualitative metrics. Our code

is publicly available at https://github.com/alzayats/UDnet.

Chapter 6 differs from the previous two chapters, Chapters 4 and 5, in that it introduces a new technique called the adaptive uncertainty distribution technique. While the research in Chapters 4 and 5 focused on using self-supervised techniques for label generation, Chapter 6 explores the use of this new technique to improve the performance of the models. The adaptive uncertainty distribution technique was adopted because it offers several advantages over traditional self-supervised techniques. For example, it allows the model to better handle uncertainty in the data by adaptively adjusting the distribution of uncertainty based on the data. This can lead to improved performance and more accurate label generation.

6.1 Introduction

One of the main challenges faced in deep learning-based underwater image enhancement is the lack of large datasets of high-quality underwater images. Unlike many other terrestrial domains, such as natural images, there are relatively few high-quality underwater images available for training deep learning models. This makes it challenging to train accurate models for enhancing underwater images. Additionally, the unique properties of water, such as refraction and absorption, make it challenging to apply image enhancement techniques that have been developed for other domains [174].

Underwater environments are challenging due to the extreme range of colour and contrast, especially when compared to images acquired in controlled environments, where a vast majority of existing underwater image enhancement algorithms were trained. When natural light enters the water from the air, it can be scattered multiple times as it travels through the water. This scattered light forms the background light that illuminates the underwater scene. In addition to the background light, the light that is directly reflected off objects in the scene also travels to the camera. The total light that is perceived by the camera is the sum of these two components: the background light and the directly reflected light. This is what creates the colours and details that we see in underwater images, see Fig. 6.1 left image. Moreover, different wavelengths of light are absorbed and scattered at different rates as they travel through water. Blue light, with its shorter wavelength, is less absorbed and scattered than other colours, which is why it can travel the longest distance through water. As a result, when we look at objects underwater, the blue light is the most dominant, giving the images a blueish tint, as can be seen in Fig. 6.1 right image.

Moreover, objects immersed in water typically appear pale and blurry, making it hard

to recognize features. Also, colours are washed out, reducing contrast and visual information. This effect is known as the *aberration of refraction* and it is due to the variation in the refractive index of water as the light passes through it.

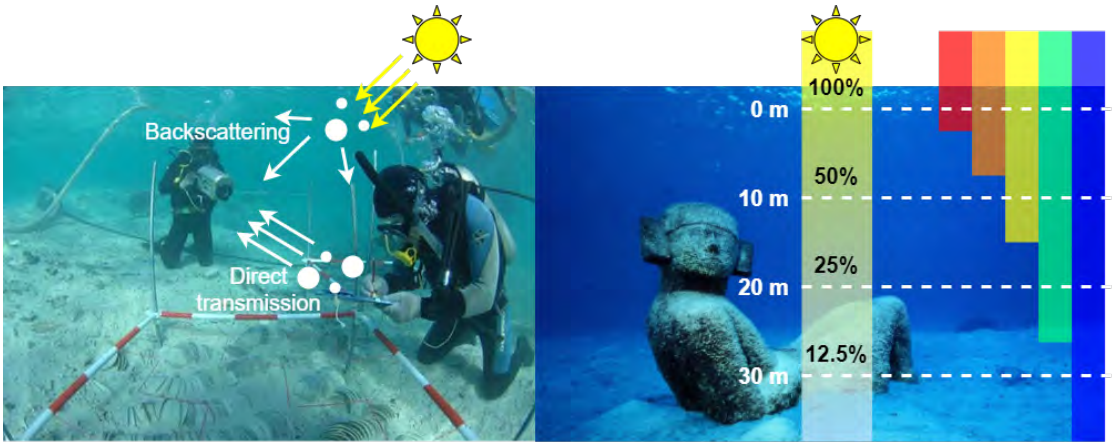

Figure 6.1: (Left) Natural light entering the water is scattered multiple times, forming the backscattering for the underwater scene. The light directly reflected off objects in the scene also travels to the camera, and the total light perceived is the sum of these two components, creating the colours and details in underwater images. (Right) Different wavelengths of light are absorbed and scattered differently as they travel through water. Blue light travels the longest distance due to its shorter wavelength, making underwater objects appear blue in colour.

When light travels from one medium to another, it bends, or refracts, due to the change in the speed of light in the new medium. This can cause the light to be focused at a different point than where it would be focused in the absence of refraction, resulting in distortion of the image. Aberration of refraction is particularly significant in underwater photography, where the refractive index of water is different from that of air. Due to the refraction, colourful underwater objects appear pale, and contrast is lost between the sea objects and backgrounds.

Furthermore, there is an imbalance between colour channels, depending on the wavelength range of light that has travelled through the water. This effect is known as *colour cast* and is typically caused by incorrect white balance, which is the process of adjusting the colours in an image so that objects that are supposed to be white are actually rendered as white. If the white balance is not correct, the colours in the image may appear unnatural or distorted.

Due to these difficulties, underwater image enhancement is a challenging problem that requires specific considerations and the development of specialized algorithms and tech-

niques. These algorithms and techniques aim to improve the quality of images taken underwater and can account for correcting distortion caused by water, improving colour accuracy, and increasing the overall contrast and sharpness of the image. There are a variety of techniques used for underwater image enhancement, including colour correction , contrast enhancement, and noise reduction.

While traditional image enhancement methods such as histogram equalization do a reasonable job of improving the visual quality of underwater images, current traditional methods struggle with the complexity of underwater images. Despite the difficulties, there is an ever-increasing need for high-quality enhancement of underwater images. For example, underwater robots require high-quality images with proper colour and contrast to accurately perform automated tasks like object detection and target recognition. For example, Fig. 6.2 shows how the underwater image enhancement model increases underwater robots' capacity to visually perceive their surroundings.

Underwater robots are often used in a variety of applications, including oceanic geological exploration, resource exploitation, ecological research, and others. However, their visual perception is often affected by environmental factors such as water clarity, light conditions, and the presence of particles or other objects in the water. These factors can make it difficult for underwater robots to accurately perceive and interpret their surroundings, which can limit their ability to perform their intended tasks. To overcome this, researchers are developing various approaches to improve the visual perception of underwater robots, such as using advanced imaging sensors and software for image enhancement and object recognition. However, specialized hardware platforms and cameras can be expensive with a high power demand, which is not suitable for underwater robots especially when they perform edge processing.

Due to its importance, underwater image enhancement has been an active area of research, despite its many challenges. Early methods relied on histogram equalization and contrast stretching and were not able to achieve effective results. Recently, much progress has been made in the field of underwater image enhancement. For instance, [373] has shown that convolutional neural network (CNN) based image enhancement algorithms perform well on underwater images, achieving enhanced images with improved contrast and colour reproduction. The work in [374] introduced a CNN-based image enhancement framework for underwater images that is able to automatically determine optimal enhancement parameters, resulting in images with both high quality and low computational cost. This method has achieved state-of-the-art performance compared to prior works in image enhancement for underwater images. However, the main limitation of all

previous methods is that they are trained in a supervised manner with a set of potential reference images (labels), which is reliant on a human to manually select the best one as the ground truth. Yet another limitation of these works is the limited high-quality and realistic underwater image datasets for their training.

To address these limitations, in this work, we explore an alternative approach to supervised underwater image enhancement. Specifically, we propose a novel unsupervised framework that removes the need for human-selected labels. The unsupervised random labels (reference map images) are generated by an SGMCSS module. This reference map generation module is an iterative multi-scale statistically guided multi-colour space stretch that improves the visual quality of the auto-generated reference image. Multi-scale statistically guided means that the stretch modules adaptively increase the contrast of images by regulating histogram distribution in distinct colour spaces. This is essential to the training process of the proposed model since it needs to both model the uncertainty and the statistical similarity between the input image and the reference image. The output of the reference map generator is used by a U-Net-like cVAE to extract a better representation of image features, which are fed to a PAdaIN block. This block can be viewed as a form of Bayesian feature learning, in which learning happens automatically through back-propagation. In contrast to prior work in underwater image enhancement [375], we use PAdaIN to transform the global enhancement statistics of input features. We also leverage it to encode uncertainty in the label image distributions, which can result in better image quality.

Our main contributions are summarized as follows:

- We propose a probabilistic unsupervised underwater image enhancement framework that leverages an encoder-decoder network that learns the uncertainty distribution of the underwater images from a pair of raw and reference images.
- To improve the visual quality of the reference image, we develop a multi-scale statistically guided multi-colour space stretch module.
- We show that while our method does not need manual human annotation it outperforms state-of-the-art supervised models on several datasets.

The rest of this paper is organized as follows. In Sec. 6.2, we review the related work on underwater image enhancement. In Sec. 6.3, we describe the architecture and different components of our proposed framework in detail. We describe the dataset and the experimental setup used in our work, while presenting our results and evaluations in Sec. 6.4.

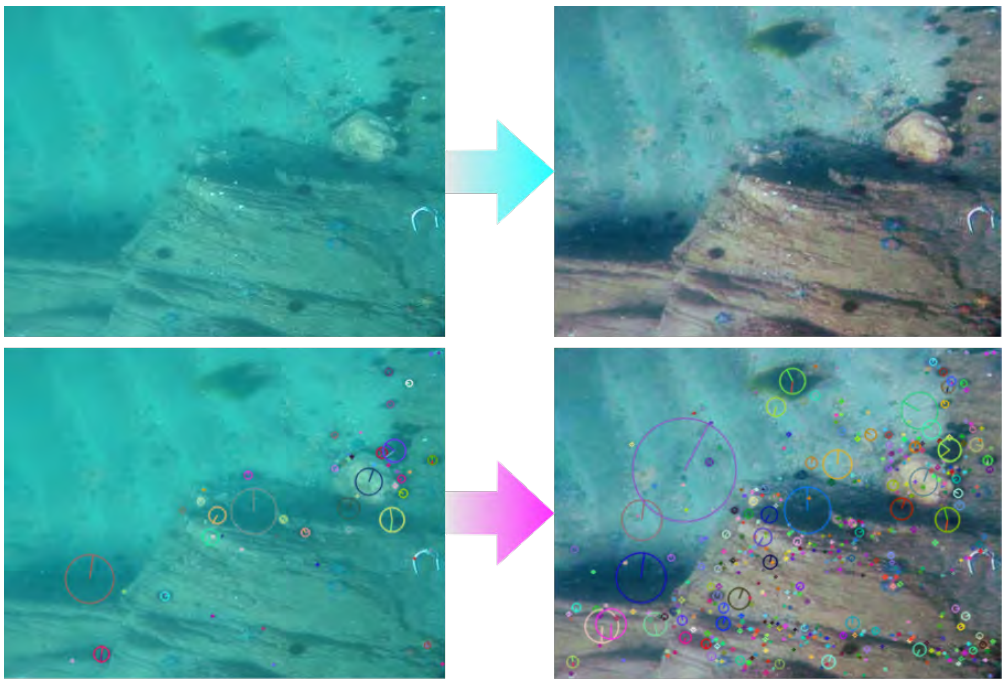

Figure 6.2: Sample enhancement results achieved using our proposed underwater image enhancement model. A typical example application of our method is increasing underwater robots' capacity to visually perceive their surroundings by improving their ability in detecting image features and key points. The right column shows images before enhancement, while the left are images after enhancement. The bottom images show key points detected using the SIFT method, before and after enhancement. Please refer to Sec. 6.4.7 for details. Best viewed online for colour and details.

The detailed discussions of our results are presented in Sec. 6.5. Finally, we provide concluding remarks and directions for further research in Sec. 6.6.

6.2 Related Work

One of the main approaches for enhancing underwater images is called prior-based, which uses a physical-model-based method to improve the accuracy of image enhancement. Physical-model-based methods aim to improve underwater images by using visual cues to estimate the optical parameters that affect the images. By applying these parameters in reverse, the original, unaltered images can be reconstructed. Examples of the visual cues used in these methods include the red channel prior [376], the underwater dark channel prior [377], and the underwater light attenuation prior [378]. One example of the use of light attenuation prior is a dehazing algorithm proposed by Chiang *et al.* [379]. Galdran *et*

al. [376] proposed another method that is a variation of the Dark Channel Prior approach that uses red channel information to calculate the depth of underwater images. Berman *et al.* [380] looked at using various spectral profiles of different water types and calculated two global parameters: the attenuation ratios of the blue-red and blue-green colour channels. Another prior-based method, called Sea-thru, was developed by Akkaynak *et al.* [381] and uses RGBD images as input to estimate backscatter and the attenuation coefficient.

Another approach to underwater image enhancement is known as the model-free method, which enhances the images without taking into account the degradation process that they have undergone. Examples of such methods include traditional contrast-limited adaptive histogram equalization , white balance which adjusts the overall colour balance of an image to compensate for the blueish tint that is often present in underwater images, and Retinex [382]. In some cases, these methods may be combined or enhanced with additional techniques, such as fusion-based approaches or multi-scale strategies , to improve their performance. In other cases, these methods may improve image quality by adjusting pixel values. One such example is [383], which proposes an unsupervised colour correction (UCM) technique. It uses white balance to even out the different colours, and then it makes the image's contrast higher. Another technique called adaptive histogram enhancement, builds on UCM and uses Rayleigh distribution stretching to make images clearer and reduce over-enhancement. These methods can make underwater images look brighter and clearer.

Deep learning-based methods are the third category of approaches to enhancing underwater images. These methods use training data to automatically extract useful information from the images and apply it to enhance it. Deep-learning-based image enhancement methods can be divided into convolutional neural networks (CNNs) and generative adversarial networks (GANs). One example of the use of CNN is the study by Sun *et al.* [384]. They proposed a CNN model that utilized an encode-decoder framework for removing noise from underwater images. Li *et al.* [385] developed a lightweight CNN model that incorporates information about the underwater scene to synthesize degraded images. They also proposed a model that uses information about light transmission to improve the quality of degraded images in specific regions. Another example is the use of a GAN proposed by Li *et al.* [386], to generate synthetic underwater images in an unsupervised manner. This approach allows for the training of an enhancement network using the synthetic data. Another example is the weakly supervised method proposed by Li *et al.* [387], which reduces the need for paired data. Additionally, Guo *et al.* [388] enhanced

degraded underwater images using a multi-scale dense GAN. Islam *et al.* [18] introduced FUnIE-GAN, a GAN model inspired by U-Net that is effective at improving colour and sharpness in underwater images.

In contrast to previous studies, our approach integrates probabilistic-based methods into deep-learning-based methods, which automatically learn the uncertainty distribution of underwater images. Specifically, our deep learning method includes conditional variational autoencoders (cVAEs). A variational autoencoder (VAE) is a type of deep learning generative model that can create new data that is similar to its training data. A VAE consists of two parts: an encoder, which maps the input data to a low-dimensional latent space, and a decoder, which maps the latent space back to the original input space. The encoder and decoder are trained together to maximize the likelihood of the generated data, while also enforcing a regularization constraint on the latent space, which encourages the model to learn a compact and meaningful representation of the input data.

Variational autoencoders (VAEs) and conditional VAEs have been widely used in computer vision tasks. Unlike traditional encoders that output a single value for each latent state attribute, VAEs formulate the encoder to describe a probability distribution for each attribute. To effectively train a VAE, both a regularizer and a reconstruction loss must be implemented. These tools penalize any inconsistencies between the VAE's posterior and prior distributions of the latent representation. VAEs and cVAEs have been used in a variety of applications related to underwater image enhancement. For instance, in some studies, VAEs were utilized to model the background of images for salient object detection , while in others they were employed to learn motion sequence generation . Some researchers have even used VAEs to denoise images and predict multiple deprojected instances of images and videos. Additionally, VAEs have been combined with contrastive learning to identify and enhance salient features. Overall, VAEs have proven to be useful for generating diverse solutions in a variety of settings.

6.3 Method

This section describes the various components and concepts utilized to build our Uncertainty Distribution Network (UDnet). As shown in Fig. 6.3, UDnet is composed of three abstract building blocks including a reference map generation block that uses a statistically guided multi-colour space stretch module, a feature extractor block that uses a cVAE, and a probabilistic adaptive instance normalization block. All of these blocks and their underlying components and concepts will be discussed in detail below, how-

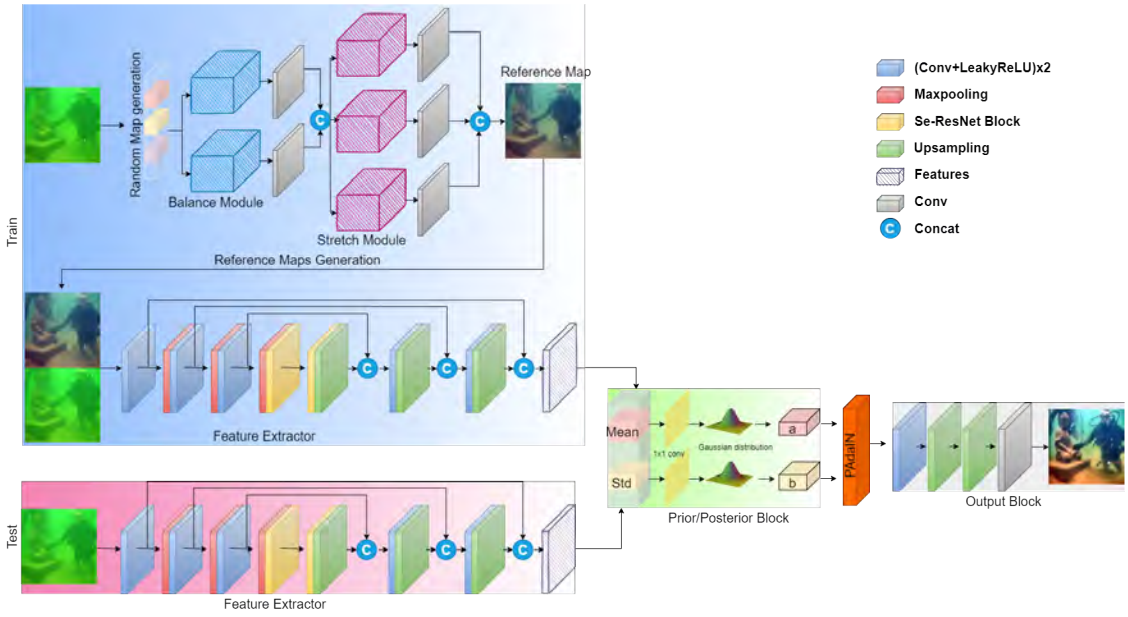

Figure 6.3: The architecture of UDnet is composed of three abstract modules: the reference maps generation module, the feature extractor module, and the PAdaIN module. The input is a three-dimensional underwater image with pixel values ranging from 0 to 1. UDnet generates a random enhanced reference map image from the input image, and uses a U-Net-based cVAT feature extractor to map input images to representations, which are then transformed by the PAdaIN module to create the enhanced image. In the training phase, the feature extractor is used to calculate the posterior distribution, and random samples from this distribution are used to transform the enhancement representation. In the testing phase, a single degraded image is used as the input and random samples from the Prior distribution are used to generate the enhanced output image. The detailed structure of each module is described in subsequent subsections of the paper.

ever, the reader is encouraged to investigate the full detail of our implementation code at https://github.com/alzayats/UDnet.

6.3.1 Reference Maps Generation

The main challenge when training deep learning networks for underwater image enhancement is the limited availability of reference maps (labels) for degraded input images. To address this issue, we auto-generated reference maps based on Underwater Image Enhancement Benchmark Dataset (UIEBD) [20], which contains real-world underwater images and corresponding reference maps generated using 12 state-of-the-art enhancement algorithms.

Autogenration of three reference maps from the input image

In the original UIEBD, volunteers were asked to compare the enhanced results and subjectively select the best one as the final reference image. However, our reference map generation process uses the same intuition without human intervention. Using the degraded input image (original image), as shown in the first step of Fig. 6.3, we generate three enhanced reference maps by three enhancement algorithms, one of which is randomly selected to introduce uncertainty into our training dataset. It is worth mentioning that adding more enhanced reference maps did not increase the model accuracy as discussed in more detail in Sec. 6.4.8.

The three methods that we chose to introduce uncertainty into the dataset were contrast and saturation adjustment, as well as gamma correction on the original images. These methods were chosen because they can effectively simulate the distortions commonly found in underwater images, such as changes in contrast, saturation, brightness, and colours. As shown in Eq. (6.1), the contrast and saturation adjustment were performed using a linear transformation formula, where the adjustment coefficient α was the same for all pixels for contrast adjustment and varied for each pixel for saturation adjustment.

$$y = (x - m) \times \alpha + x, \tag{6.1}$$

where x and y refer to the degraded and enhanced images, respectively, m denotes the mean of each channel, and α is the adjustment coefficient.

Our approach has several advantages, including saving time and increasing reliability compared to using human observers to generate reference maps. We evaluated the effectiveness of the generated reference maps in Sec. 6.4.5 and Sec. 6.4.6 by comparing the enhanced results to the subjective selections made by volunteers in the original UIEBD dataset. Our goal was to create uncertain labels that would reflect the uncertainty in the ground truth recording, rather than significantly altering the original labels. To achieve this, we utilized a Statistically Guided Multi-Colour Space Stretch (SGMCSS) or Colour Correction module.

Statistically Guided Multi-Colour Space Stretch for Colour Correction

The colour correction module's goal is to improve the colour and contrast of the reference maps generated. This is obtained by transforming the reference map Red Green Blue (RGB) values to the optimal RGB values, which involves determining the proper

camera white-balance for colour-neutral subjects, as well as removing the effects of lens flare and red-green chromatic aberration. This could be useful when dealing with over-saturated images. The colour correction module is designed for the case where the mean and standard deviation of the red green and blue colour values are known. This module uses a non-parametric approach to colour correction [374], which is able to accommodate new statistical distributions of the pixel values in the red, green and blue colour channels. The colour correction module consists of two main components: a dual-statistic balance module and a multi-colour space stretch module.

In the dual-statistic balance module, the image is processed by two different modules that use statistics of the image (average and maximum values) to correct its colour balance. The output is then enhanced using two residual-enhancement modules to recover lost details.

The first residual-enhancement module is based on Grey World (GW) theory. The Gray World theory is a method for colour correction in images. It is based on the assumption that the average colour of objects in a perfect image is grey, which means that the average values of the R, G, and B channels are equal. This means that the scale factors for each channel, e_R, e_G, and e_B, can be determined using the GW theory:

$$x^{GW} = Conv_{1\times1}(x) \otimes \overline{A},$$

where $\overline{A} = [\frac{1}{A_R}, \frac{1}{A_G}, \frac{1}{A_B}] \in \mathbb{R}^{3\times1}$, A_c denotes the average value of c channel in the original image, and $\otimes$ denotes pixel-wise multiplication.

The second residual-enhancement module is based on the White Patch (WP) algorithm. The White Patch algorithm is another method for colour correction in images. It is based on the assumption that the maximum response of the RGB channels in an image is caused by a white patch in the scene. This white patch is assumed to reflect the colour of the light in the scene, so the largest value in the RGB channels is used as the source of light. Based on this hypothesis, the scale factors for each channel can be expressed as:

$$x^{WP} = Conv_{1\times1}(x) \otimes \overline{M},$$

where $\overline{M} = [\frac{1}{M_R}, \frac{1}{M_G}, \frac{1}{M_B}] \in \mathbb{R}^{3\times1}$, M_c denotes the maximum value of c channel in original image..

The two residual-enhancement results are merged and passed to the stretch module as follows:

$$x^{DSB} = Conv_{3\times3}(x^{GW}) \oplus Conv_{3\times3}(x^{WP}),$$

where x^{DSB} represents the result enhanced by the dual-statistic balance module.

In the multi-colour space stretch module, the image is transformed into different

colour spaces (HSI and Lab) and processed by a trainable module to improve contrast. The original image is also enhanced and added to the stretched version as follows:

$$x^{final} = Conv_{3\times3}(x^r) \oplus Conv_{3\times3}(x^h) \oplus Conv_{3\times3}(x^l),$$

Where x^r, x^h, x^l denote the histogram stretched pixel value in RGB, HSI, and Lab colour spaces, respectively.

The H channel of the HSI colour space is preserved without any changes. The output is then converted back to the RGB colour space and merged together by going through 3×3 convolutional layer and pixel-wise add up. Overall, this technique can improve the visual quality of the reference map that will be passed to the next building block of UDNet, i.e. the feature extractor module (see Fig. 6.3), by correcting colour balance and enhancing contrast.

6.3.2 Feature Extraction

The next abstract building block of UDNet, as shown in Fig. 6.3, is its feature extractor block. UDnet uses a two-branch U-Net-based feature extractor to map the input images to representations. These representations are then fed into the PAdaIN module, which transforms the enhancement statistics of the input to create the enhanced image as explained in 6.3.3.

The training branch of the feature extractor is used to construct posterior distributions using the raw original underwater image and its corresponding reference map image as inputs. The test branch, on the other hand, is used to estimate the prior distribution of a single raw underwater image.

The PAdaIN block is used to encode the uncertainty in the input image, allowing UDNet to generate multiple enhanced versions of the image that capture the different possible interpretations of the original image. To achieve this, UDnet uses a prior/posterior block to build the distribution of possible enhancements. This block is designed to construct both a mean and a standard deviation distribution, using 1×1 convolutions to transform the input data matrix into a series of distributions that capture the uncertainty in the input image.

In the training stage, the input image and its corresponding reference image are used to learn the *posterior* distributions of the latent codes as follows:

$$\boldsymbol{a} \sim \mathcal{N}_m\left(\boldsymbol{\mu}\left(\boldsymbol{y}, \mathbf{x}\right), \boldsymbol{\sigma}^2\left(\boldsymbol{y}, \mathbf{x}\right)\right), \tag{6.2}$$

$$\boldsymbol{b} \sim \mathcal{N}_s \left(\boldsymbol{m} \left(\boldsymbol{y}, \mathbf{x} \right), \boldsymbol{v}^2 \left(\boldsymbol{y}, \mathbf{x} \right) \right), \tag{6.3}$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are two random samples from the mean and standard deviation posterior distributions, $\mathcal{N}_m$ and $\mathcal{N}_s$ are the N-dimensional Gaussian distribution of the mean and standard deviation, and $\boldsymbol{y}$ and $\boldsymbol{x}$ are the reference image and the raw input image, respectively.

Once these distributions have been constructed, random samples are extracted from them and injected into the PAdaIN module, where they are used to transform the statistics of the received features.

In the testing stage, the latent codes generated for PAdaIN are determined only by the input image to learn the *prior* distributions of the latent codes as follows:

$$\boldsymbol{a} \sim \mathcal{N}_m \left(\boldsymbol{\mu} \left(\mathbf{x} \right), \boldsymbol{\sigma}^2 \left(\mathbf{x} \right) \right), \tag{6.4}$$

$$\boldsymbol{b} \sim \mathcal{N}_s \left(\boldsymbol{m} \left(\mathbf{x} \right), \boldsymbol{v}^2 \left(\mathbf{x} \right) \right), \tag{6.5}$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are two random samples from the mean and standard deviation prior distributions, $\mathcal{N}_m$ and $\mathcal{N}_s$ are the N-dimensional Gaussian distribution of the mean and standard deviation, respectively and $\boldsymbol{x}$ is the raw input image.

The UDnet model is applied multiple times to the same input image in order to generate multiple enhancement variants. This is done by re-evaluating only the PAdaIN module and the output block, without retraining the entire model, which makes UDNet very efficient. The resulting diverse enhancement samples are then used for Maximum Probability estimation that takes the enhancement sample with the maximum probability as the final estimation.

Loss Function

The training process for UDnet follows the standard procedure for training a cVAE model, which involves minimizing the variational lower bound. However, our approach has an additional step of finding a meaningful embedding of enhancement statistics in the latent space. This is achieved through the use of a posterior network (as shown in Fig. 6.3), which learns to recognize posterior features and map them to posterior distributions of the mean and standard deviation. Random samples from these distributions can be used to formalize the enhanced results. This approach allows for the incorporation of uncertainty into the enhancement process, which can improve the accuracy and reliability of the resulting images.

During the training process, the PAdaIN module is used to predict the enhanced image by receiving random samples $\boldsymbol{a}$ and $\boldsymbol{b}$ from Eq. (6.2) and Eq. (6.3), respectively. The enhancement loss (Eq. (6.6)) is calculated based on the differences between the predicted image and the reference map, and is used to penalize the model if the output deviates from the reference.

$$L_e = L_{mse} + \lambda L_{vgg16}, \tag{6.6}$$

where L_{mse} denotes the mean square error loss and L_{vgg16} denotes the perceptual loss [389], λ refers to a weight parameter.

The mean square error loss L_{mse} and the perceptual loss L_{vgg16} are two common metrics used to evaluate the performance of image enhancement algorithms. The mean square error loss measures the average squared difference between the predicted and reference images, while the perceptual loss, which was introduced by Johnson et al. [389], measures the differences between the high-level features of the predicted and reference images. The weight λ is used to control the relative importance of these two loss terms in the overall enhancement loss L_e. For example, if λ is set to a high value, the model will be more heavily penalized for large differences between the predicted and reference images, while if λ is set to a low value, the model will be less sensitive to such differences. The specific values of λ used in the training process will depend on the characteristics of the dataset and the desired performance of the model.

In addition to minimizing the enhancement loss L_e, the training process for UDnet also involves using Kullback-Leibler (KL) divergences D_{KL} to align the posterior distributions with the prior distributions (Eq. (6.7) and Eq. (6.8)).

$$L_m = D_{KL}\left(\mathcal{N}_m\left(\mathbf{x}\right) \| \mathcal{N}_m\left(\boldsymbol{y}, \mathbf{x}\right)\right), \tag{6.7}$$

$$L_s = D_{KL}\left(\mathcal{N}_s\left(\mathbf{x}\right) \| \mathcal{N}_s\left(\boldsymbol{y}, \mathbf{x}\right)\right), \tag{6.8}$$

where $\boldsymbol{m}$ and $\boldsymbol{s}$ are the mean and the standard deviation, respectively. KL divergence is a measure of the difference between two probability distributions and can be used to compare the posterior distributions learned by the model with the prior distributions that are assumed to represent the distribution of latent variables in the training data. By minimizing the KL divergences between the posterior and prior distributions, the model is able to learn a more accurate representation of the latent space, which can improve the quality of the enhanced images.

The total loss function used for training UDnet is the weighted sum of the enhancement

loss L_e and the KL divergences D_{KL} between the posterior and prior distributions,

$$L = L_e + \beta(L_m + L_s), \tag{6.9}$$

where β is a weight parameter, whose value depends on the dataset's characteristics and the model's desired performance. By minimizing this total loss function, L, the model is able to learn an effective mapping from the input degraded images to the corresponding enhanced images, while also aligning the posterior and prior distributions in the latent space. This allows the model to generate high-quality enhanced images while also incorporating uncertainty into the enhancement process.

6.3.3 Probabilistic Adaptive Instance Normalization (PAdaIN)

The final abstract building block of UDNet, as shown in Fig. 6.3 is PAdaIN block. The goal of UDnet is to adjust the appearance of underwater images, such as the colours and contrasts, without altering the content of the image. This is important because it allows the enhanced images to be more visually appealing and easier to interpret, without compromising the integrity of the original image. Therefore, We use a probabilistic adaptive instance normalization (PAdaIN) to capture these properties.

The proposed PAdaIN method [375] is based on the AdaIN algorithm [390], which is commonly used for style transfer in image processing. AdaIN adjusts the mean and standard deviation of the features of the content image to match those of the style image, effectively changing the appearance of the content image without altering its content. This approach is useful for underwater image enhancement because it allows the model to adjust the appearance of the image without changing its content, which is important for maintaining the integrity of the original image.

However, AdaIN relies on the availability of known content and style images, which is not always the case in underwater image enhancement processes. To address this issue, PAdaIN introduces random samples from the posterior distributions of the mean and standard deviation as the parameters of the AdaIN operation, which can be formulated as:

$$PAdaIN\left(\mathbf{x}\right) = \boldsymbol{b}\left(\frac{\mathbf{x} - \boldsymbol{\mu}\left(\mathbf{x}\right)}{\boldsymbol{\sigma}\left(\mathbf{x}\right)}\right) + \boldsymbol{a}, \tag{6.10}$$

where $\boldsymbol{b}$ and $\boldsymbol{a}$ are two random samples from the posterior distributions of the mean and standard deviation, respectively.

These posterior distributions are learned using a cVAE, which was described in 6.3.2.

This allows PAdaIN to generalize the AdaIN algorithm and apply it to underwater image enhancement without the need for known content and style images. Overall, PAdaIN is able to capture the important appearance-related features of the input image and use them to generate enhanced images that maintain the integrity of the original image.

It is worth noting that in contrast to other approaches that consider the variance of the image, such as GAN, PAdaIN is based on the statistical distribution of the image features, which are invariant to transformations like colour transformation. This is done by conditioning the network on training images and their reference map, which, along with the use of a differentiable approximation of the uncertainty, make UDnet easily trainable with a single backward pass.

6.3.4 Uncertainty Distribution

Overall, through its novel structure, UDNet learns to adapt to uncertainty distribution. Here, uncertainty distribution refers to the inherent uncertainty that can exist in the image enhancement process, because different images need different types of enhancements such as contrast, saturation, gamma, or other enhancements. The main idea behind UDnet is to better incorporate this uncertainty in the enhancement process. This is motivated by the fact that the true clean image is often unavailable, and that there is a degree of uncertainty in the labels used to train the image enhancement models. Existing deterministic learning-guided methods [391] are unable to capture this uncertainty, and therefore, have to make compromises between different possible enhancement results.

To address this issue, UDnet uses an implicit variable $\mathbf{z}$ to represent the uncertainty in the enhancement process. This variable could represent human subjective preferences, or the parameters of the camera or enhancement algorithms used to capture or generate the ground truth images, which could affect the outcome of the enhancement process. By taking this uncertainty into account, UDnet is able to more accurately capture the range of possible enhancements, rather than trying to determine a single "correct" result. This is particularly useful in situations where the true, unaltered image is not available or cannot be accurately reproduced.

The goal of UDNet is to learn a mapping from the low-quality input image $\mathbf{x}$ to the clean image $\mathbf{y}$ that takes into account the uncertainty represented by $\mathbf{z}$. This can be formalized as follows:

$$p\left(y\left|\mathbf{x}\right.\right) \approx p\left(\mathbf{y}\left|\mathbf{z}_{\max}, \mathbf{x}\right.\right), \mathbf{z}_{\max} \sim p\left(\mathbf{z}\left|\mathbf{x}\right.\right), \tag{6.11}$$

where $p(\mathbf{z}|\mathbf{x})$ denotes the distribution of uncertainty, and $\mathbf{z}_{\max}$ denotes the sample with the maximum probability.

Eq. (6.11) represents the probabilistic framework underlying UDnet. In this equation, $p(\mathbf{y}|\mathbf{z}_{\max}, \mathbf{x})$ is the probability of the clean image $\mathbf{y}$ given the sample with the maximum probability $\mathbf{z}_{\max}$ and the low-quality input observation $\mathbf{x}$. $p(\mathbf{z}|\mathbf{x})$ is the probability of the uncertainty variable given the observation. The goal of the model is to learn these probability distributions from the training data and then use them to generate enhanced images that incorporate uncertainty into the enhancement process. By doing so, UDnet is able to (1) provide users with multiple alternative enhancement results to choose from, or (2) improve the accuracy and reliability of the final enhancement result by taking the enhancement sample with the maximum probability as the final estimation, without user intervention.

6.4 Experiments

In this section, we perform several experiments to evaluate the performance of our proposed method. We will first describe the utilized datasets, evaluation metrics and implementation details. Then, we quantitatively and qualitatively evaluate our model against 10 popular image enhancement models on 8 public datasets. Finally, we will demonstrate the significance of our work through a visual perception improvement test.

6.4.1 Datasets

We used eight publicly available datasets for our model's performance verification. These datasets are: EUVP [18], UFO [19], UIEBD [20], DeepFish [24], FISHTRAC [25], FishID [26], RUIE [27], SUIM [28]. Details of these datasets can be found in Table 6.1. In EUVP [18], UFO [19], and UIEBD [20], there are many paired images and unpaired images which were divided as shown in Table 6.1. The paired images are the ones that have ground truth. The rest of the datasets have only unpaired images. In our experiment, we used only UIEBD [20] for training in an unsupervised way without the ground truth. We used the other datasets for performance evaluation.

6.4.2 Evaluation Metrics

Evaluation metrics for image enhancement are often based on natural image statistics. Perceptual and structural image qualities can be judged in different ways. We employ

Table 6.1: The datasets used in our research. The numbers represent the number of images in sets

Datasets	Train		Test	
	Paired	Unpaired	Paired	Unpaired
EUVP [18]	3700	3140	515	-
UFO [19]	1500	-	120	-
UIEBD [20]	800	-	90	60
DeepFish [24]	-	3200	-	600
FISHTRAC [25]	-	600	-	71
FishID [26]	-	7093	-	6897
RUIE [27]	-	2904	-	726
SUIM [28]	-	1525	-	110

four full-reference evaluation metrics and three no-reference evaluation metrics for evaluating the quantitative performance of our image enhancement model. Specifically, 1) The full-reference evaluation metrics consist of Peak Signal-to-Noise Ratio (PSNR) [21], Structural Similarity (SSIM) [21], Most Apparent Distortion (MAD) [22], and Gradient Magnitude Similarity Deviation (GMSD) [23], which are used for paired test sets (EUVP [18], UFO [19], UIEBD [20]). A higher PSNR or a lower MAD score means that the output image and the label image are closer in perceptual content, while a higher SSIM or a lower GMSD score means that the two images are more structurally similar. 2) The no-reference evaluation metrics are: Underwater Image Quality Measure (UIQM) [29], Multi-scale Image Quality Transformer (MUSIQ) [30], and Natural Image Quality Evaluator (NIQE) [31] which are used for unpaired test sets (DeepFish [24], FISHTRAC [25], FishID [26], RUIE [27], SUIM [28]). The UIQM is the linear combination of three underwater image attribute measures: the underwater image colourfulness measure (UICM), the underwater image sharpness measure (UISM), and the underwater image contrast measure (UIConM). A higher UIQM and MUSIQ or a lower NIQE score suggests a better human visual perception. However, it is worth noting that these no-reference metrics cannot accurately reflect the quality of an image in some cases [392], so scores of UIQM, MUSIQ, and NIQE are only provided as references for our study. We will present enhanced unpaired images in the visual comparisons section for readers to appraise.

Table 6.2: Comparison against published works on three *paired* datasets (EUVP [18], UFO [19] and UIEBD [20]). underwater image enhancement performance metric in terms of average PSNR [21], SSIM [21], MAD [22] and GMSD [23] values are shown, where (↑) means higher is better and (↓) means lower is better. We represent the best two results in RED and BLUE colours. * the model trained on [20] dataset with label.

Method	EUVP				UFO				UIEBD			
	PSNR ↑	SSIM ↑	MAD ↓	GMSD ↓	PSNR ↑	SSIM ↑	MAD ↓	GMSD ↓	PSNR ↑	SSIM ↑	MAD ↓	GMSD ↓
CLAHE	18.97	0.726	138.6	0.090	18.76	0.701	143.7	0.098	20.64	0.821	100.9	0.053
IBLA	22.62	0.719	97.78	0.068	20.71	0.671	122.6	0.082	17.56	0.614	141.2	0.126
PIFM*	20.17	0.747	113.5	0.0719	20.63	0.728	118.2	0.076	23.62	0.852	80.80	0.056
PUIEnet*	21.01	0.770	94.55	0.052	21.38	0.737	102.3	0.057	23.74	0.844	79.36	0.057
RGHS	21.13	0.753	98.40	0.056	20.74	0.730	112.8	0.066	23.57	0.803	81.02	0.053
UCM	20.91	0.767	99.37	0.062	20.34	0.743	110.0	0.068	22.03	0.815	92.95	0.067
UDCP	15.80	0.572	136.8	0.098	15.95	0.561	148.1	0.111	13.47	0.548	139.0	0.118
ULAP	21.91	0.730	108.4	0.071	21.98	0.729	116.7	0.071	18.95	0.718	113.0	0.085
USLN*	20.87	0.771	94.62	0.050	20.73	0.749	105.0	0.057	24.04	0.849	78.91	0.057
Wavenet*	20.25	0.753	109.3	0.067	20.98	0.736	115.1	0.071	24.61	0.881	68.08	0.045
UDnet (ours)	22.96	0.771	87.67	0.049	22.43	0.738	99.53	0.053	22.23	0.812	74.21	0.043

Table 6.3: Comparison against published works on five *unpaired* datasets (DeepFish [24], FISHTRAC [25], FishID [26], RUIE [27], and SUIM [28]). underwater image enhancement performance metric in terms of average UIQM [29], MUSIQ [30] and NIQE [31] values are shown, where (↑) means higher is better, and (↓) means lower is better. We represent the best two results in RED and BLUE colours.

Method	DeepFish			FISHTRAC			FishID			RUIE			SUIM		
	UIQM↑	MUSIQ↑	NIQE↓	UIQM↑	MUSIQ↑	NIQE↓	UIQM↑	MUSIQ↑	NIQE↓	UIQM↑	MUSIQ↑	NIQE↓	UIQM↑	MUSIQ↑	NIQE↓
CLAHE	3.136	25.67	4.10	2.686	48.94	3.78	2.631	37.01	5.20	3.028	34.98	4.48	2.914	58.40	3.66
IBLA	1.993	43.34	6.31	1.704	55.36	6.20	1.745	44.45	6.69	2.577	32.60	4.68	1.839	58.17	4.10
PIFM	3.275	27.39	4.29	2.892	48.38	4.17	2.424	41.14	5.12	3.087	33.29	4.51	2.694	58.56	3.82
PUIEnet	3.209	28.82	4.10	3.421	48.60	4.22	2.301	40.08	5.32	3.102	28.32	4.56	2.838	60.03	3.75
RGHS	3.150	42.26	6.72	1.913	57.03	5.57	1.823	44.46	5.95	2.991	30.49	4.32	2.317	59.55	3.75
UCM	2.918	41.37	6.45	2.575	54.39	11.25	2.452	44.32	6.06	3.107	32.59	4.19	2.804	58.51	3.90
UDCP	2.391	42.42	5.72	1.622	51.01	6.09	1.320	37.97	6.48	2.159	29.79	4.71	1.731	56.14	4.09
ULAP	2.814	40.46	6.16	1.763	54.37	6.18	2.176	45.12	5.98	2.396	33.00	4.72	2.232	58.36	3.97
USLN	3.015	32.12	4.24	3.316	49.97	4.30	2.067	44.58	6.32	3.068	32.78	4.44	2.682	61.30	4.03
Wavenet	3.304	40.14	3.00	3.071	56.79	3.70	2.252	46.66	5.15	3.081	30.79	4.56	2.773	61.03	3.67
UDnet(ours)	3.292	24.56	4.02	3.390	50.24	3.41	2.303	37.85	5.11	3.154	26.74	4.18	2.875	57.84	3.65

Table 6.4: Comparison against published works on two *paired* datasets (UIEBD [20], and EUVP [18]) and two *unpaired* datasets (UCCS [27], and UIQS [27]) .
Underwater image enhancement performance metric in terms of average PSNR [21], SSIM [21], UIQM [29], UCIQE [29], higher values is better we represent the best two results in red and blue colours.

Method	UIEBD				EUVP				UCCS		UIQS	
	PSNR	SSIM	UIQM	UCIQE	PSNR	SSIM	UIQM	UCIQE	UIQM	UCIQE	UIQM	UCIQE
FIRUA	27.80	0.849	3.452	0.534	20.65	0.728	2.985	0.316	3.650	0.652	3.223	0.809
RCA-CycleGAN	21.28	0.804	2.987	0.280	21.322	0.829	2.834	0.337	2.942	0.671	2.975	0.832
IEFD	22.01	0.793	3.234	0.317	21.17	0.705	3.148	0.363	3.199	0.598	3.379	0.827
RFHP	20.31	0.841	3.189	0.598	20.91	0.649	2.932	0.299	3.345	0.627	3.788	0.801
Two-step DA	21.78	0.802	3.402	0.624	20.37	0.711	3.105	0.208	3.724	0.674	3.418	0.913
MCACE	20.69	0.784	3.293	0.587	20.89	0.698	2.927	0.309	3.933	0.587	2.851	0.856
Twin ACL	22.30	0.888	3.595	0.683	21.04	0.724	3.029	0.354	3.953	0.688	3.032	0.897
UDnet(ours)	22.23	0.812	3.781	0.745	22.96	0.771	3.265	0.749	3.974	0.713	3.154	0.958

6.4.3 Implementation Details

Our models were trained with an input resolution of 256×256 pixels. We scale the lowest side of the image to 256 and then extract random crops of size 256×256. We found that for this problem set, a learning rate of 1×10^{-4} works the best. It took around 500 epochs for the model to train on this problem and the batch size was set as 10. Our networks were trained on a Linux host with a single NVidia GeForce RTX 2080 Ti GPU with 11 GB of memory, using Pytorch framework. The training is carried out with ADAM optimizer, and the loss function, as explained in 6.3.2, is a combination of the Mean Squared Error (MSE) L_{mse}, the perceptual loss L_{vgg16} [389], and Kullback-Leibler (KL) divergences L_{kl}.

In order to boost network generalisation, we augment the training data with rotation, flipping horizontally and vertically. Following [375], we adopt 1×1 convolutions to broadcast the samples to the desired number of channels before input to PAdaIN with a latent space of a 20-dimensional N.

6.4.4 Compared Methods

To have a comprehensive and fair evaluation of our model, we compare it to 10 previous studies including six conventional unsupervised methods (CLAHE [393], IBLA [394], RGHS [395], UCM [383], UDCP [377], ULAP [378]) and four deep-learning-based methods (PIFM [373], PUIEnet [375], USLN [374], Wavenet [391]). The comparison with conventional unsupervised methods aims to demonstrate the advantages of our trainable unsupervised deep-learning-based method.

We applied these conventional unsupervised approaches directly to the test sets. We used the respective studies' code and training approach for the deep learning-based methods. To guarantee the experiment's objectivity, we trained the four deep-learning-based methods on UIEBD [20] and applied the author-provided model and network training parameters.

6.4.5 Quantitative Comparisons

The comparison results for all paired test sets are summarized in Table 6.2. We report the average scores of the four full-reference metrics (PSNR, SSIM, MAD, GMSD). Table 6.2 demonstrates that our proposed method outperforms all six conventional unsupervised methods and all four deep-learning-based methods in all four full-reference metrics on

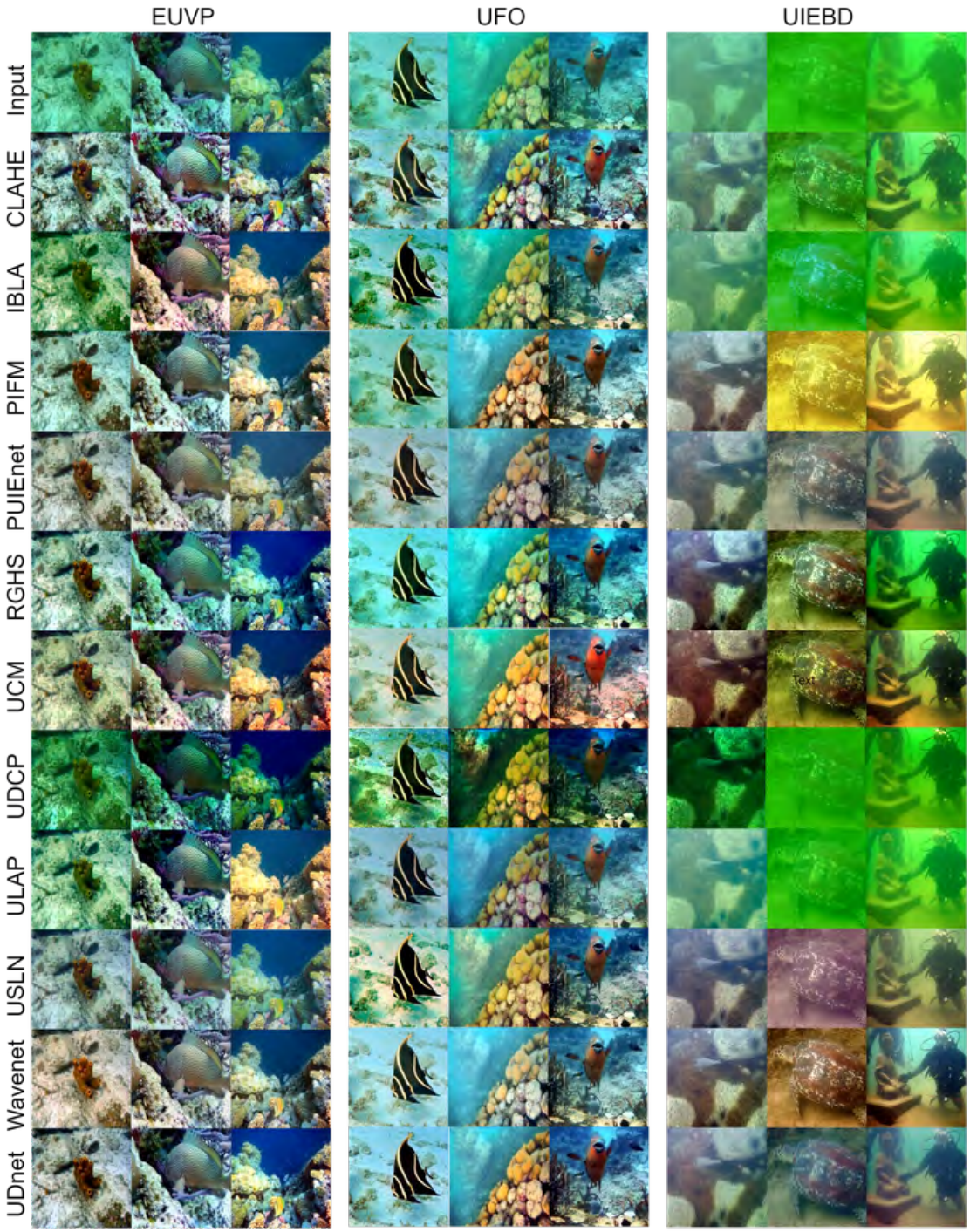

Figure 6.4: Visual comparisons on challenging underwater images sampled from paired datasets, i.e. EUVP [18], UFO [19], and UIEBD [20]. The name on the right of each row refers to the enhancement method used.

the EUVP dataset and shows great performance on the UFO dataset. Our model achieves the highest PSNR, SSIM scores on EUVP, and the lowest MAD, GMSD scores on EUVP,

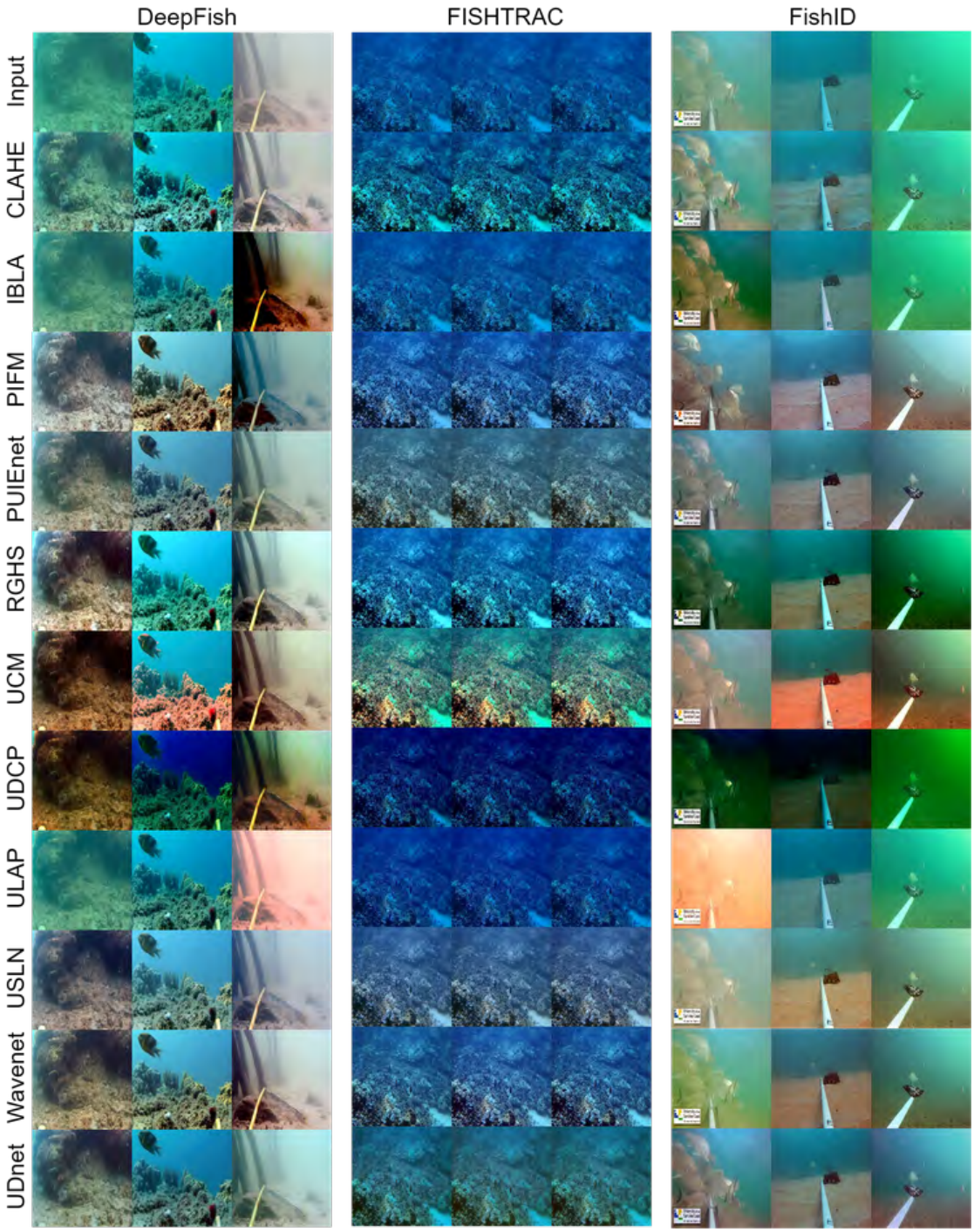

Figure 6.5: Visual comparisons on challenging underwater images sampled from DeepFish [24], FISHTRAC [25], and FishID [26]. The name on the right of each row refers to the method. We also include a short video of our model's prediction at https://youtu.be/k4ASsGze5p8 and https://youtu.be/NV5GH-GG_3c.

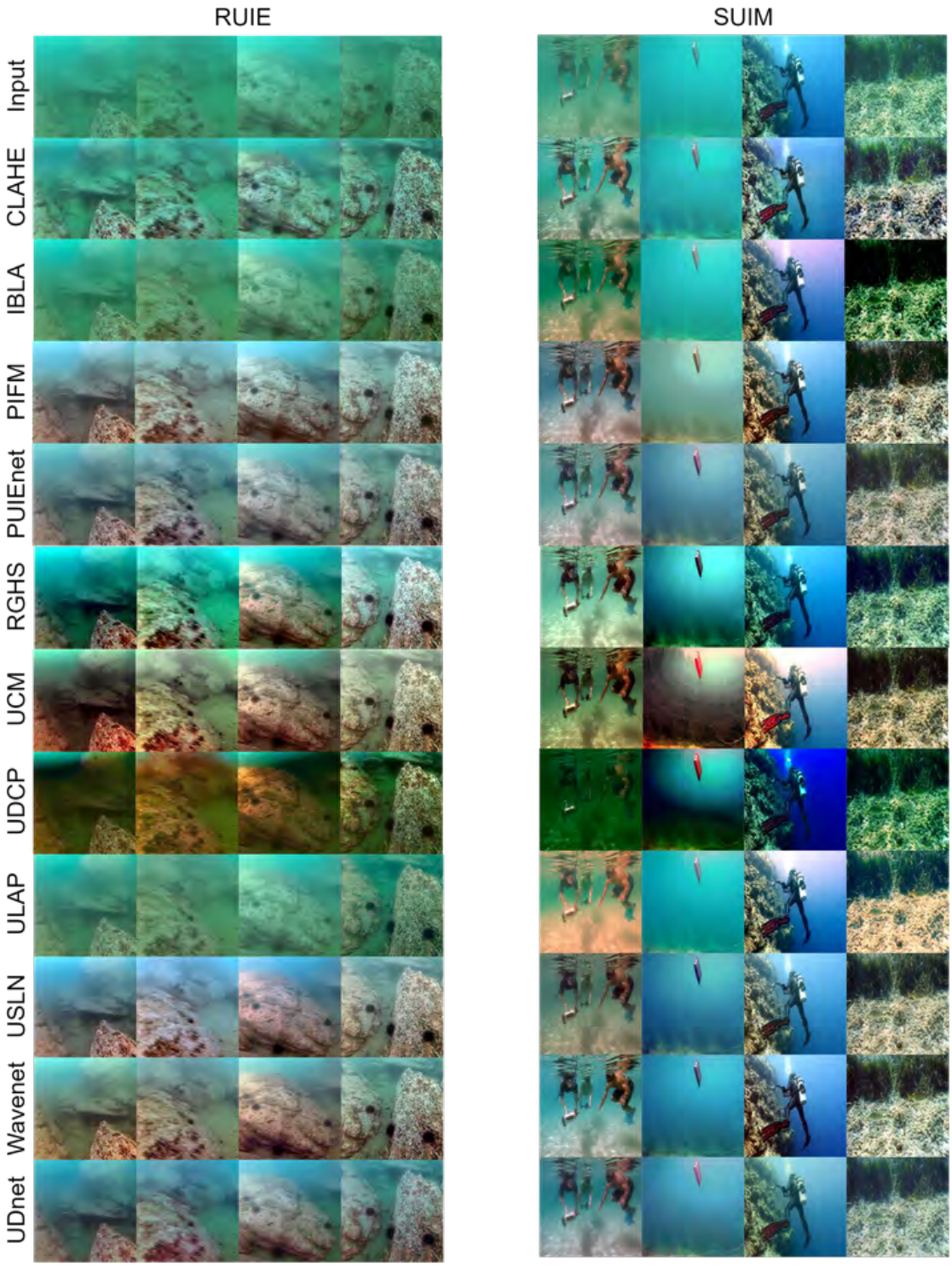

Figure 6.6: Visual comparisons on challenging underwater images sampled from RUIE [27], and SUIM [28]. The name on the right of each row refers to the method.

UFO. In addition, we also found that

- Although our model was trained in an unsupervised way, it still outperformed the fully supervised deep-learning-based models trained on UIEBD dataset on EUVP and UFO.
- This result shows that our proposed unsupervised deep-learning-based method is better than conventional ones at preserving structural information and contrast preservation, which suggests the superiority of our trainable model.
- Our model's performance is slightly lower than fully supervised methods on UIEBD dataset. However, this is because fully supervised methods were trained on the images and ground truth labels acquired from the UIEBD.
- Even without any extra labels, our model outperforms models that are trained on the UIEBD dataset on the two metrics of MAD and GMSD for paired datasets.

We also provided quantitative comparisons for unpaired test sets in Table 6.3, which demonstrate that our model achieves the highest NIQE on FISHTRAC, FishID, RUIE, SUIM, and the second-best UIQM score on DeepFish, FISHTRAC, SUIM. These results also show that

- Deep-learning-based models cannot outperform conventional approaches in no-reference evaluation metrics, in contrast to full-reference evaluation metrics.
- The quantitative results suggest that our method can generalize well on unseen datasets even without ground truth.

According to the results presented in Table 6.4, our proposed method, UDnet, demonstrates superior performance in underwater image enhancement when compared to other published works on four datasets: UIEBD, EUVP, UCCS and UIQS. Specifically, UDnet achieved the highest average values for PSNR, SSIM, UIQM and UCIQE metrics on these datasets. For instance, on the UIEBD dataset, UDnet achieved the highest UIQM and UCIQE values while on the EUVP dataset, it achieved the highest PSNR, UIQM and UCIQE values. Similarly, on the UCCS dataset, UDnet achieved the highest UIQM and UCIQE values and on the UIQS dataset, it achieved the highest UCIQE value. These results indicate that UDnet is a robust method for enhancing underwater images.

Our method, UDnet, achieves better performance than other methods due to the effective combination of several components. The cVAE module is able to learn a compact and

informative representation of the underwater image content, which helps to preserve important details during the enhancement process. The PAdaIN module is able to adaptively adjust the style of the enhanced image to match the target domain, resulting in more natural and visually pleasing results. Finally, the multi-colour space stretch module is able to effectively enhance the contrast and colour of the underwater images by stretching the colour histogram in multiple colour spaces. These components work together to produce high-quality enhanced underwater images.

6.4.6 Qualitative Comparisons

Underwater images possess several unique characteristics. They have more texture content and low luminance and contrast compared to terrestrial images. Therefore, it is important to assess human visual perception in terms of image content enhancement in underwater images, especially in terms of colour enhancement. In order to gain more insights into the effectiveness of our proposed UDnet, we performed comprehensive investigations and comparisons among all eight datasets using the ten previous methods introduced.

Fig. 6.4 demonstrates three example raw input images of each of the three paired datasets in the first row, along with the enhanced image outputs from the 10 aforementioned studies and our UDNet. This comparison has a two-fold purpose:

- To demonstrate the effectiveness of the deep-learning-based methods in the no-reference settings.
- To showcase the superiority of our unsupervised method, which has enhanced the underwater scenes without ground truth for training.

Furthermore, to prove the superiority of our model in handling unpaired images, we show visual comparisons of randomly selected underwater images from the five aforementioned unpaired datasets in Fig. 6.5, and Fig. 6.6. We also include a short video of our model's prediction at https://youtu.be/k4ASsGze5p8 and https://youtu.be/NV5GH-GG_3c.

As Fig. 6.5 shows, the obvious light limitation of the raw image results in low contrast. For example, UDCP and ULAP models tend to make the image darker, while others such as UCMeven introduce reddish colour. In comparison, our model increases both brightness and contrast, making the details of the image clear. The input image samples given in Fig. 6.6 mostly suffer from obvious green deviation, which cannot be resolved

by most models. For example, CLAHE, IBLA, and ULAP fail to remove the green deviation. In comparison, our model removes the greenish colour and makes the image colours balanced.

Overall, our qualitative comparison results show:

- Even when the ground truth label of the paired images is added to enhance visual quality, some of the previous methods show problems such as over-enhancement, lack of contrast, and saturation.
- Some of the models' output images from the paired dataset have over- or under-enhanced backgrounds, while some have no change in the background. However, the output image of our model does not show such problems.
- Some of the models' output images' background pixels are saturated. However, our model has not suffered from the over- or under-saturation problem.

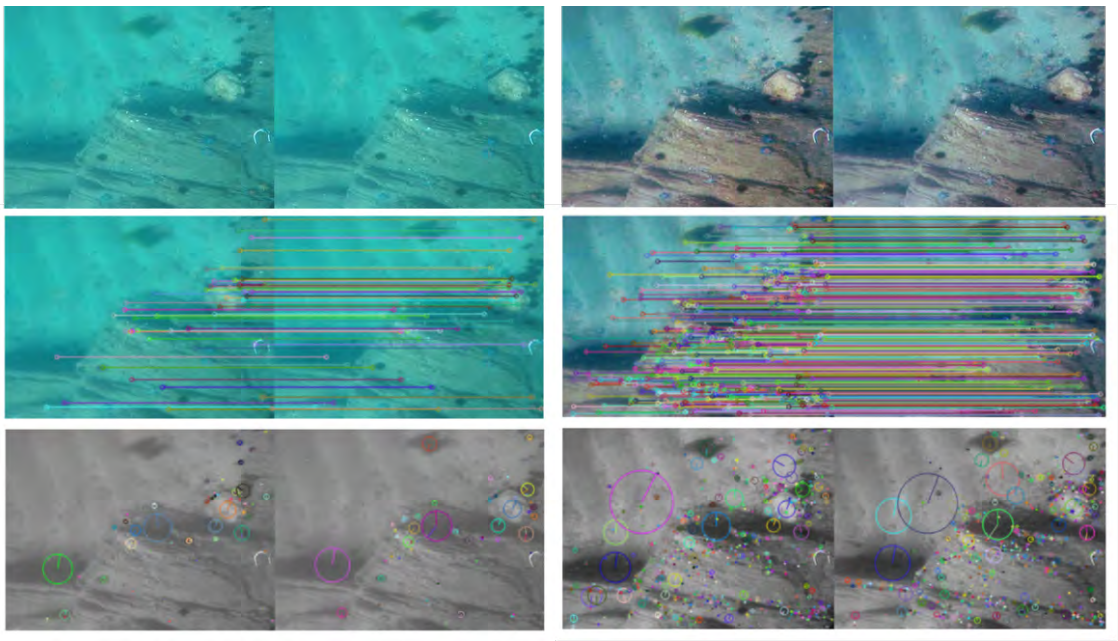

Figure 6.7: Comparison of image feature and key points matching before (left) and after (right) image enhancement with our model. From the top: the original images, matched feature points, and SIFT keypoints. The images are from RUIE [27] dataset.

Table 6.5: Ablation study: comparison against different model variants on UIEBD dataset in terms of average PSNR and SSIM values

Model variant	PSNR ↑	SSIM ↑
w/o colour	22.01	0.791
All colour	21.73	0.784
Multi-label	22.12	0.795
w/o VGG	21.89	0.789
Full Model	22.23	0.812

6.4.7 Visual Perception Improvement

One of the main objectives of underwater image enhancement is to increase underwater robots' capacity to visually perceive their surroundings. This is essential for robots to make autonomous decisions in complex underwater scenarios. To evaluate our model's performance in visual perception improvement, we used feature detection and matching to assess its capability in improving the visual perception of underwater images. Feature detection and matching are commonly used techniques in many computer vision applications, such as structure-from-motion, image retrieval, object detection, and image stitching. Here, we use Scale-Invariant Feature Transform (SIFT), which helps locate the local features in an image (keypoints), and Random Sample Consensus (RANSAC) [396], which is used to match feature points. These methods are used to compare the visual perception of an underwater image before and after enhancement. Fig. 6.7 depicts the result for two consecutive frames from RUIE [27] dataset, (blue_01.jpg) and (blue_02.jpg). These show that the numbers of matched points between the two image frames increase from 74 (before the enhancement) to 594 after the enhancement. At the same time, the number of SIFT keypoints also dramatically increases as a result of the enhancement, significantly improving the visual perception of the environment.

6.4.8 Ablation Study

For a more in-depth analysis of the proposed method, we analyzed the effect of its different components and stages including its colour correction module, adding extra reference maps to the original three maps, and VGG loss. The quantitative comparisons are presented in Table 6.5, where

- **w/o colour** means that UDnet is trained without the colour correction module.

- **All colour** means that UDnet is trained with a colour correction module in all inputs, not just the reference map generation stage.
- **Multi-label** means that UDnet is trained with 6 extra generated enhanced reference maps.
- **w/o VGG** means that UDnet is trained without VGG loss.

We used PSNR, and SSIM to evaluate the results on UIEBD, which are shown in Table 6.5. Compared to the ablated model variants, the full model achieves the best quantitative performance on both metrics. The qualitative comparisons are presented in Fig. 6.8. Overall, the conclusions drawn from the ablation study are:

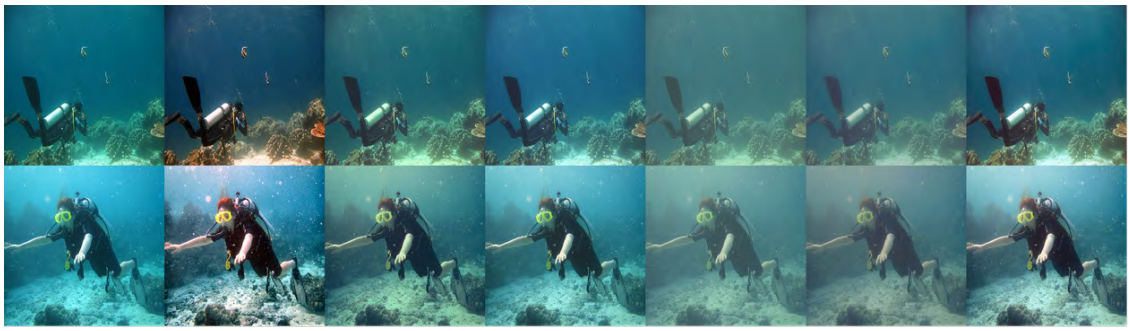

Figure 6.8: ABLATION STUDY: The qualitative comparison of the contributions of multiple stages of the proposed framework on the UIEBD dataset. (a) Input, (b) ground truth, (c) w/o colour, (d) All colour, (e) Multi-label, (f) w/o VGG, (g) Full Model.

1. Without the colour correction module, the proposed model generates unsatisfactory results (see Fig. 6.8(c)). The enhanced image still has low contrast. For example, it has no contrast at the bottom of the image and in the green part, which both belong to the sea bed region.
2. With the colour module in all inputs, the model performs better, see Fig. 6.8(d). The image is not as dark as w/o the colour module. However, the model still underenhances the detailed information at the bottom of the image and the green parts.
3. Adding more reference maps to the original three generated by contrast and saturation adjustment, and gamma correction,does not improve the quality of the output images (see Fig. 6.8(e)).

4. When UDnet is trained without VGG loss, the quality of the generated image is degraded (see Fig. 6.8(f)). The reduced performance as presented in Table 6.5 demonstrates that the VGG loss can make the model learn more information about the original underwater scene from the labels and improve the visual quality of output.

6.5 Discussion

A variety of deep learning-based techniques have been proposed to enhance underwater images. Most of these techniques are based on supervised learning that requires a large amount of manually labeled ground truth images. In this work, we propose a novel unsupervised underwater image enhancement framework, which leverages an encoder-decoder network that works by adaptively enhancing the input image with the statistical information of a randomly selected reference image at train time. This means that we leverage the probabilistically enhanced reference image to obtain a more robust enhancement of that image. This enables the network to adaptively enhance the input image at training time using a stochastic approach.

The potential applications of underwater image enhancements are broad and varied. For example, improved accuracy and reliability of underwater image enhancement could be beneficial in a range of fields, including:

- Environmental monitoring: Better-quality underwater images could be used to track changes in marine ecosystems, such as coral reefs, and monitor the health of aquatic species.
- Marine biology: Enhanced underwater images could provide more detailed information about the behaviour and habitats of marine animals, which could be useful for research and conservation efforts.
- Underwater archaeology: Improved image quality could make it easier to identify and study artifacts and structures in underwater archaeological sites.

In addition to these applications, our proposed UDNet can be used for automatically generating reference maps for training other models for underwater image enhancement. Furthermore, our approach could be extended to other domains where reference maps are difficult to obtain, such as medical imaging [8] or satellite image enhancement. In medical imaging, for example, reference maps are often difficult to obtain due to the complexity and variability of human anatomy. Automatically generated reference maps could help

to improve the accuracy and reliability of medical image enhancement, which could be beneficial for diagnostic and therapeutic purposes. Similarly, in the field of satellite image enhancement, reference maps are often difficult to obtain due to the large spatial and temporal scales involved. Automatically generated reference maps could improve the accuracy and reliability of satellite image enhancement techniques, which could be useful for applications such as environmental monitoring and disaster response.

6.6 Conclusion

In this work, we presented a novel unsupervised deep learning approach for underwater image enhancement, which is a challenging problem due to the random distortion and low contrast of underwater images. Our proposed UDNet framework leverages an encoder-decoder network that adaptively enhances the input image with the statistical information of a randomly selected reference image during training. The results of our experiments show that our approach outperforms ten popular underwater image enhancement methods in seven common metrics, both for paired and unpaired input images across eight public datasets. This demonstrates the effectiveness of our proposed approach in enhancing underwater images.

One of the strengths of our approach is its unsupervised learning nature, which enables it to be applied in situations where ground truth data is not available. The model's strong generalization ability in handling unpaired datasets, as demonstrated in this study, is a testament to its robustness. Additionally, the proposed UDNet has the potential to be applied in a variety of fields, including environmental monitoring, marine biology, and underwater archaeology, which can greatly benefit from the improved accuracy and reliability of underwater images.

However, our model also has some limitations. One of the difficulties in enhancing underwater images is the presence of backscatter, particularly at far distances. Although we have used state-of-the-art image enhancement algorithms to process raw underwater images, backscatter can still be present in some cases, which affects the performance of our model. Additionally, some existing algorithms use inaccurate image formation models or assumptions that limit their performance in enhancing underwater images.

In future work, we plan to address these limitations by exploring the effectiveness of other CNN architectures in the underwater image enhancement problem. We also plan to improve the visual quality of enhanced images and the accuracy of reference maps by using a multi-resolution approach, which can help capture different types of features in

the underwater environment at various resolutions. This can lead to better overall image quality and improved capture of the nuances of the underwater environment.

In conclusion, our proposed unsupervised deep learning approach for underwater image enhancement provides a significant improvement in enhancing underwater images. The framework's strong generalization ability, its potential applications in various fields, and its unsupervised learning nature make it an attractive solution for researchers in the field to apply to the challenging problem of underwater image enhancement. However, there is still room for improvement, and future work will focus on addressing the limitations of the current approach and improving the visual quality of enhanced images.

Chapter 7

MFLD-net: A Lightweight Deep Learning Network for Fish Morphometry using Landmark Detection

Monitoring the morphological traits of farmed fish is pivotal in understanding growth, estimating yield, artificial breeding, and population-based investigations. Currently, morphology measurements mostly happen manually and sometimes in conjunction with individual fish imaging, which is a time-consuming and expensive procedure. In addition, extracting useful information such as fish yield and detecting small variations due to growth or deformities, require extra offline processing of the manually collected images and data. DL and specifically Convolutional Neural Networks (CNNs) have previously demonstrated great promise in estimating fish features such as weight and length from images. However, their use for extracting fish morphological traits through detecting fish keypoints (landmarks) has not been fully explored. In this Chapter, the third research question is addressed. Specifically, we developed a novel DL architecture that we call Mobile Fish Landmark Detection network (MFLD-net). We show that MFLD-net can achieve keypoint detection accuracies on par or even better than some of the state-of-the-art CNNs on a fish image dataset. MFLD-net uses convolution operations based on Vision Transformers (i.e. Patch embeddings, Multi-Layer Perceptrons). We show that MFLD-net can achieve competitive or better results in low data regimes while being lightweight and therefore suitable for embedded and mobile devices. We also provide quantitative and qualitative results that demonstrate its generalisation capabilities. These features make MFLD-net suitable for future deployment in fish farms and fish harvesting plants.

7.1 Introduction

Morphology is an important metric in the production of farmed fish because it can be used to determine the weight and overall size of a fish. These variables are key to animal health and welfare and are used for phenotype analyses in advanced breeding programs. In aquaculture, determining the morphology is a frequent task crucial for selecting fish for culture as well as developing and testing novel fish strains. Furthermore, fish morphological traits are valuable resources for artificial breeding [397], functional gene mapping [398], and population-based investigations [399]. Morphology helps identify when fish are mature enough to produce eggs or sperm. When determining the growth and maturity of fish, a number of specific morphological traits may be evaluated including the distance from the tip of the mouth to the posterior midpoint of the caudal fin, or the depth of the body from the posterior base of the dorsal fin to anterior of the anal fin [35]. However, traditional manual fish morphology measurement methods are inefficient and time-consuming. A typical fish measuring process includes measuring the fish's weight using a digital scale, measuring its body lengths with a ruler and then recording these values. Not only is this process inefficient and labour-intensive, but it is also prone to human error.

An automatic tool can help aquaculturists and animal health and welfare authorities to save time and reduce costs by quickly characterising fish morphology and predicting their overall quality in a fast, accurate, and cost-effective manner. Furthermore, the tool would improve the quality of information available to fish farmers and may unlock niche information, because the system can be used at scale. A promising technique to automate this measurement process is computer vision used along with machine learning to capture fish images and automatically extract fish morphology.

A few previous works [400–402] have used computer vision and traditional image processing techniques to segment [400, 402] or make a 3D model [401] of the fish body to then use classical machine learning methods e.g. regression [400, 401] for weight and/or length extraction. Although these studies have achieved significant results, they have involved a complex image processing and feature engineering process to suit their experimental conditions. In contrast, more recent research has been motivated by the outstanding performance of Deep Learning (DL)-based Convolutional Neural Networks (CNNs) in processing images, without the need for complex image processing and/or feature engineering steps [198, 321, 403]. In [198, 321], the authors have used a CNNs to predict fish body weight by feeding fish images to a segmentation CNN to extract the fish-body area. These studies have utilised both the entire fish body (*i.e.* fish outline) and excluded

fins and tails for weight estimation through mass-area estimation models. In [403], the authors have also explored the use of CNN for estimating the weight and length of fish but they have utilised different CNN architectures. Specifically, the authors have implemented a SegNet-like CNN architecture with an image resolution of 512×512 to estimate the correlation between body measurements and body weight, carcass weight, and carcass yield.

Additionally, researchers have used CNNs for predicting morphological characteristics such as overall length and body size by detecting keypoints on the fish body [404, 405], similar to the proposed method in this research. However, [405] has proposed a CNN classifier to detect only two keypoints, the fish head, and tail fork regions, to measure the fish body length. On the other hand, [404] has used two neural networks, i.e. a Faster R-CNN [236] to first detect the fish in the image and then a Stacked Hourglass [406] to detect specific keypoints on the initially-detected fish, which makes the proposed method complex and expensive. In a more recent study, [407] proposed a CNN for marine animal segmentation with good results on a self-curated dataset. However, the 207.5 million trainable parameters of their network make it unsuitable for usage in embedded systems or on mobile computing devices for easy deployment in fish farms. To the best of our knowledge, previously published studies that use deep learning to predict fish's morphological traits are complex and large. This renders them unsuitable for use within embedded and mobile devices for commercial use at scale and for easy integration into fish farms. This is because these devices, which are usually designed for resource-constrained environments, have limited computational and power budgets making them incapable of running large networks such as the one proposed in [407]. To address the lack of a lightweight but efficient fish morphological measurement tool, we develop a new Deep Learning (DL) model for fish body landmark detection using CNNs.

CNNs have dominated the design of DL systems used for computer vision tasks for many years. However, architectures based on emerging Transformer models, such as Vision Transformer (ViT) [290], have been shown to outperform standard convolutional networks in many of these tasks, especially when large training datasets are available. These recent advances motivated us to explore transformer-based architectures for developing lightweight but efficient fish landmark detection networks for automatic fish morphometric analyses.

Vaswani *et al.* first [304] suggested transformers for machine translation, and they have subsequently become the standard solution for many Natural Language Processing (NLP) applications. Since then, there have been several attempts to incorporate convolu-

tional network characteristics into transformers, making the Vision Transformers (ViT). Recently, the use of patch embeddings for the first layer of the network has spawned a new paradigm of "isotropic" designs, i.e., those having identical sizes and shapes across the network. These models resemble repeating transformer's encoder blocks, but instead of self-attention and Multi-Layer Perceptron (MLP) operations, alternative operations are used. For example, Bello *et al.* [408] introduced a two-dimensional relative self-attention mechanism replacing convolutions as a stand-alone computational primitive for image classification. ResMLP [409] built upon multi-layer perceptrons for image classification by a simple residual network that alternates between a linear layer and a two-layer feed-forward network.

Because of its capacity to capture long-distance interactions, self-attention has been widely adopted as a computational module for modelling sequences [410]. For example, Ramachandran *et al.* [411] replaced all instances of spatial convolutions with a form of self-attention applied to a CNN model to produce a fully self-attentional model that outperforms the baseline on ImageNet classification.

Inspired by the strong performance of Vision Transformers, we investigated utilizing some of ViT's architectures using convolution operations. Specifically, we studied the use of Patch Embeddings [290], Multi-Layer Perceptrons (MLP-Mixer) [412], and Isometric architectures [413]. In order to apply a transformer to greater image sizes, patch embeddings aggregate together small areas of the image into single input features. Then, MLP-Mixer works directly with the patches as input, separating the mixing of spatial and channel dimensions, while keeping the network's size and resolution constant (i.e Isometric). In this work, we utilize these techniques to modify a standard CNN's architecture to a simple model that is similar in spirit to the ViT using convolutions operations, but does not need a pre-trained model and can generalise well when trained on a small dataset.

Our proposed network, which we named MFLD-net is implemented to estimate landmarks (keypoints) on the fish body to better understand and estimate its morphology. MFLD-net can assist ecologists and fisheries managers with the fast, efficient, accurate, and non-invasive prediction of the size and other morphological aspects of the fish. This provides them with the capacity to make informed management decisions. To evaluate our model, we use an image dataset of Barramundi (*Lates calcarifer*), also known as Asian seabass. We also compare our results to several baseline models to show the performance of MFLD-net.

In summary, the contributions of this work are as follows:

1. We propose a simple CNN network that estimates the position of known keypoints

in a fixed-size fish image.

2. Due to our architectural innovations, our proposed model is fast and compact, while requiring small training data. These make our system suitable for deployment in aquaculture farms.

3. We compare our results with several baselines, including U-net [32], ResNet-18 [226], ShuffleNet-v2 [414], MobileNet-v2 [415], and SqueezeNet [416].

4. We provide an evaluation of our model on 60% of our fish image dataset to quantify its generalisation and robustness.

The rest of the paper is organized as follows. Sec. 7.2 presents our method for training and evaluating our model. Our model's framework is described in detail in Sec. 7.2.1. The experimental setup and results are presented in Sec. 7.3, while detailed discussions of our results are presented in Sec. 7.5. Finally, Sec. 7.6 concludes our paper.

7.2 Materials and methods

We ran three main experiments to test and optimize our proposed model. First, we trained our network (MFLD-net) on only 40% of our dataset. Next, we tested its predictive performance on the dataset test subset described below. Finally, we compared our MFLD-net to five models from [32, 226, 414–416]. We assessed both the inference speed and prediction accuracy of each model as well as their training time and generalisability. When comparing these models we incorporated the number of model parameters, the model size on the hard disk, and the model image throughput per second. We applied the same configuration for each of the six investigated models in order to hold the training routine the same for all models. The models are also trained using the same data augmentations, without affecting their performance.

Figure 7.1 shows a high-level flow diagram that outlines the key steps involved in our proposed method. The flow diagram consists of eight main steps: data collection, annotation, data splitting, model development, training, validation, testing, and evaluation. The flow diagram illustrates how we developed and tested our novel deep learning network for fish morphometry using landmark detection. The following sections describe in detail the materials and methods used in this work.

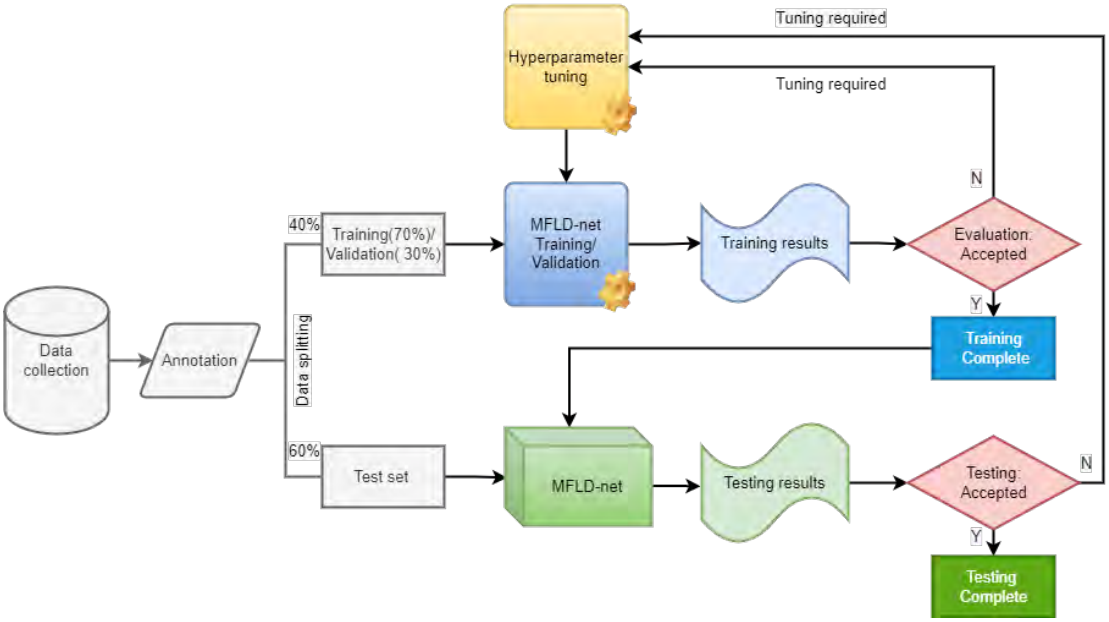

Figure 7.1: A flow diagram that outlines the key steps involved in our proposed method. The flow diagram consists of eight main steps: Data collection: Collection of fish images using a high-performance CMOS industrial camera. Annotation: Manual annotation of the images for 16 keypoints per fish. Data splitting: Random splitting of the annotated dataset into training and validation sets (70% and 30% respectively) and a test set (60%). Model development: Development of the Mobile Fish Landmark Detection network (MFLD-net) using convolution operations based on Vision Transformers, including patch embeddings and multi-layer perceptrons. Training: Training of the MFLD-net on the training set. Validation: Validation of the MFLD-net on the validation set to ensure that it is not overfitting to the training set. Testing: Testing of the MFLD-net on the test set and comparison of its performance to five other state-of-the-art baseline models. Evaluation: Evaluation of the MFLD-net's performance, including detection accuracies and generalization capabilities.

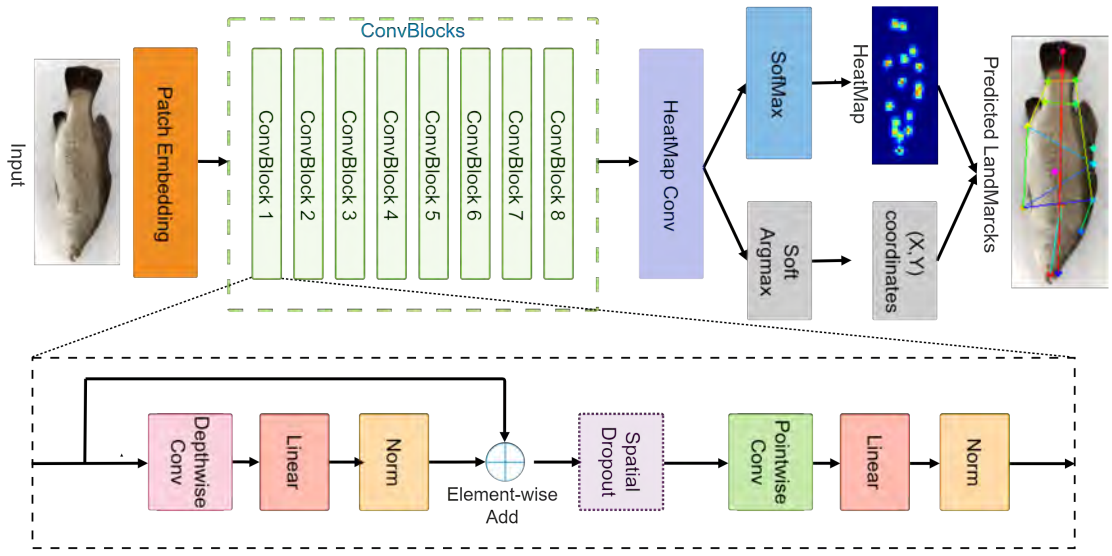

Figure 7.2: Proposed MFLD-net architecture, which is similar in spirit to the ViTs, but uses convolutions operations for keypoints estimation.

7.2.1 Model Architecture

We propose the Mobile fish landmark detection network (MFLD-net), a novel end-to-end keypoint estimation model designed as a lightweight architecture for mobile devices. We apply the architecture to address some of the main issues of current methods such as accuracy and efficiency on mobile and static keypoint estimation. The detailed architecture of MFLD-net is shown in figure 7.2. It builds upon Convolutional Neural Networks (CNNs) [413], Vision Transformer architecture [290], and Multi-Layer Perceptrons (MLP-Mixer) [412]. Additionally, MFLD-net adapts a hybrid method for processing confidence maps and coordinates that provides accurate detection for estimating keypoint locations.

To achieve higher robustness and efficiency, our architecture leverages the use of patch embedding [290], spatial/channel locations mixing [412], as well as a combination of CNNs that have the same size and shape throughout the network, i.e. are Isometric [413].

Isometric Architecture

Our model architecture is based on Isometric Convolutional Networks [413], which are made up of several similar blocks with the same resolution across the model. Architectures that are ”Isometric” have the same size and shape throughout the network and maintain a fixed internal resolution throughout their entire depth (see figure 7.2).

Sandler *et al.* [413] have demonstrated that the resolution of the input picture has only a minimal impact on the prediction quality of modern CNNs. Instead, the trade-off between accuracy and the number of multiply-adds required by the model is mostly determined by the internal resolution of intermediate tensors. Also, model accuracy can be improved further without the use of additional parameters given a fixed input resolution.

Therefore, our model has two main attributes: (1) No pooling layers while still maintaining a high receptive field. (2) Isometric networks have a high degree of accuracy while needing relatively little inference memory. These attributes make our model lightweight, hence suitable for edge processing on mobile and low power devices, such as drones and robots, which are commonplace across various industries ranging from agriculture [153] to marine sciences [417]. This lightweight design does not, however, compromise accuracy due to its use of an isometric architecture. In an era of mobile processing [174], there is a significant need for lightweight, yet powerful and effective keypoint estimation models.

Patch Embedding

Inspired by the Vision Transformer architecture [290], we experiment with applying patch embeddings directly to a standard CNN. To do so, we divide an image into patches and feed a CNN tensor layout patch embeddings to preserve locality. In an NLP application, image patches are processed similarly to tokens (words). Patch embeddings enable all downsampling to occur simultaneously, lowering the internal resolution and therefore increasing the effective receptive field size, making it simpler to combine sparse spatial information. The key advantage of using CNN instead of Transformer is the inductive bias of convolution [418, 419] such as translation equivariance and locality. Therefore, CNN is well-suited to vision tasks because it generalises well when trained on a small dataset. We implemented Patch embeddings as convolution with 3 input channels, 256 output channels, kernel size of 4, and stride of 4, followed by 8 ConvBlocks, as can be seen in fig. 7.2.

ConvBlock

Our architecture is made of 8 ConvBlocks, each consisting of depthwise convolution (i.e. mixing spatial information) as in Multi-Layer Perceptrons (MLP-Mixer) [412], and Spatial Dropout [420] for strongly correlated pixels, followed by pointwise convolution (i.e. mixing the per-location features). Each of the convolutions is followed by Gaussian error

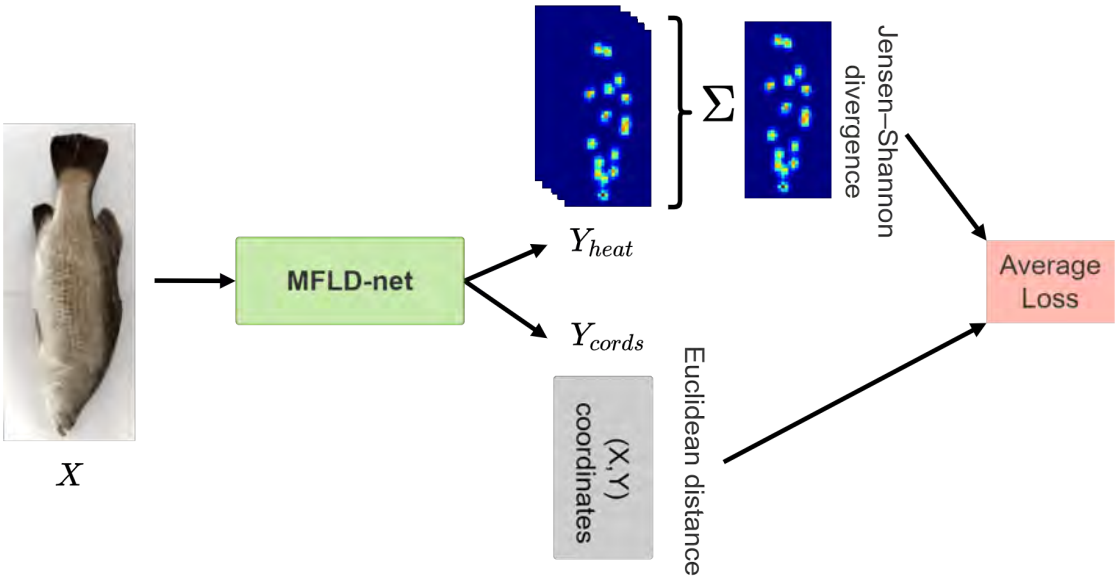

Figure 7.3: A schematic diagram of the multi-task loss function used for training MFLD-net.

linear units (GELU) [421] activation and BatchNorm. We found that for the task of keypoint estimation, architectures with a fewer number of layers result in better performance. We also added residual connections between Conv layers. We use a dropout with a rate of 0.2 to prevent overfitting. The structure of our ConvBlock can be seen in the bottom panel of fig. 7.2.

Hybrid Prediction and a Multi-Task Loss Function

Fully Convolutional Networks (FCNs) are good at transforming one image to produce another related image, or a set of images while preserving spatial information. Therefore, for our keypoint task, instead of using FCN to directly predict a numerical value of each keypoint coordinate as an output (i.e. regressing images to coordinate values), we modified FCN to predict a stack of output heatmaps (i.e. confidence maps), one for each keypoint. The position of each keypoint is indicated by a single, two-dimensional, symmetric Gaussian in each heatmap in the output, and the scalar value of the peak reflects the prediction's confidence score.

Moreover, our network not only predicts heatmaps but also predicts scalar values for coordinates of each keypoint. Therefore, during the training process, we have a multi-task loss function, which consists of two losses, i.e. JensenShannon divergence for heatmaps

and Euclidean distance for coordinates (see figure 7.3). The first loss measures the distances between the predicted heatmaps and the ground-truth heatmaps, while the second loss measures the distances between the predicted coordinates and the ground-truth coordinates. Then, we take the average of the two losses as the optimisation loss.

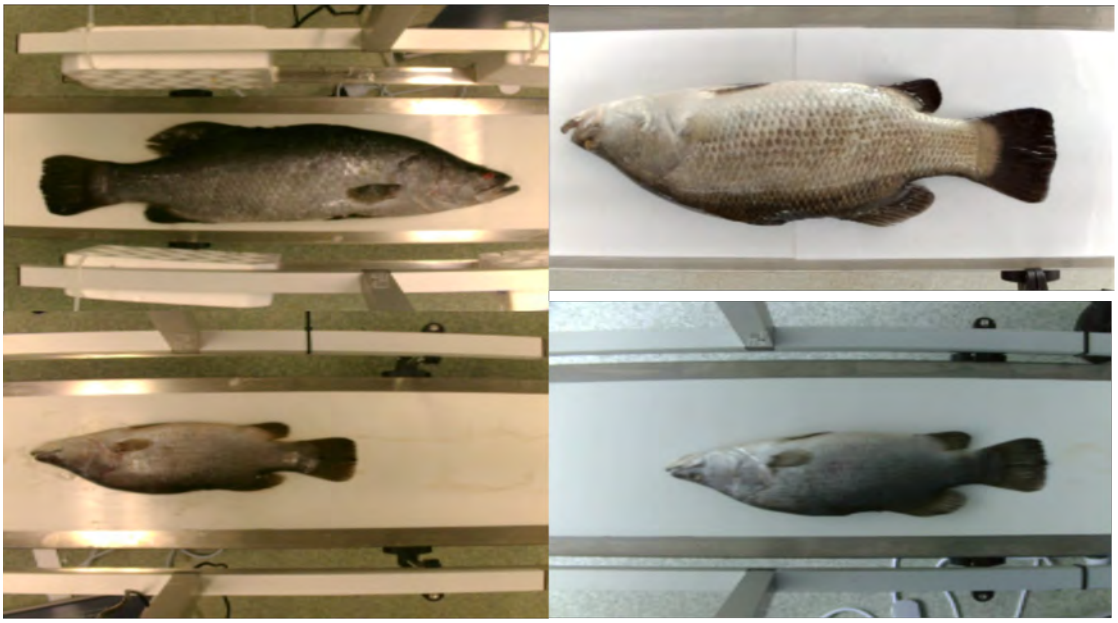

Figure 7.4: Sample images from the four data collection sessions, which are all used in our experiments.

7.2.2 Datasets

We performed experiments using a dataset of Barramundi (*Lates calcarifer*), also known as Asian seabass. These fish were photographed in a laboratory setting. The dataset was collected in four data collection sessions using the same experimental data collection setup but under four different environmental, i.e. lighting, conditions. Figure 7.4 demonstrates a sample image from each of the four data collection trials. In total, 2500 images were collected, each of which was photographed on a conveyor belt with normal ambient lighting. The images were recorded from above using a High-performance CMOS industrial camera (see figure 7.4). All barramundi were provided by the aquaculture team from James Cook University, Townsville, Australia.

To demonstrate the robustness of our network, we trained and validated our network on only 40% of the dataset. This training subset was further split into randomly selected training and validation sets, with 70% training examples and 30% validation examples.

The other 60% of the collected dataset was used only for testing the model and comparing its performance to five other state-of-the-art baseline models [32, 226, 414–416].. The images were manually annotated for 16 kepoints as shown in figure 7.5-middle. For each fish, ground truth keypoints have the form $[(x_1, y_1), ..., (x_k, y_k)]$, where (x_i, y_i) represents the ith keypoint location. Each ground truth object also has a scale s which we define as the square root of the object segment area. For each fish, our developed keypoint detector model outputs keypoint locations (see figure 7.5-right). Predicted keypoints for each fish have the same form as the ground truth, i.e. $[x1, y1, ..., xk, yk]$.

We recognize that the imaging setup used in our study was in a laboratory setting and may not fully represent the conditions of a real-world fishery. Factors such as lighting, background, and fish movement may vary significantly between a laboratory setting and a fishery. We chose to use a laboratory setting for our data collection to have a controlled and consistent environment, which reduces noise and variability in the images and improves the quality of the data for effective model development. This setup also facilitates the important annotation process, which requires manual labelling of 16 keypoints for each fish image to train the deep learning models.

We acknowledge that using a laboratory setting for our data collection has some limitations. One of the main challenges is transferring our model to a fishery setting, where the imaging conditions may vary significantly from our laboratory setup. For example, a fishery setting may have different lighting conditions, backgrounds and environments, as well as different fish species, sizes and shapes. These factors may affect the performance and accuracy of our model and the baseline models.

To address these issues and make our model as generalisable as possible, we used four different lighting conditions for our data collection, which simulate some of the variations that may occur in a fishery setting. In addition, we collected data from various fish sizes and in different orientations to augment our data collection. Furthermore, all our data was collected using a high-performance CMOS industrial camera, which is a common choice for other monitoring activities at fisheries [422].

To deploy our model to real-world fisheries setting, one approach is to perform site-specific model tuning using our baseline MFLD-Net. This means that, before deployment to a new setting, we collect some new data and retrain our model to adjust it to the new environment as well as task conditions. This adjustment is much faster and more efficient than developing a new model for the new setting. The newly added data can diversify the model's generalisation capabilities and gradually improve its performance in a wider set of environments. To perform this adjustment quicker, one approach is to use self-

supervised learning techniques to shorten the time required for a large amount of data labelling, and to add more data from other sources to improve our model.

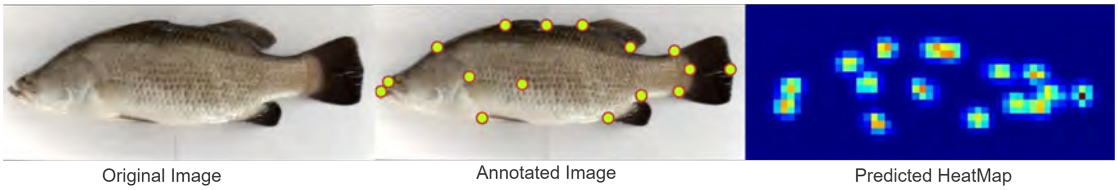

Figure 7.5: Point annotations in a sample fish image ($\mathbf{X}$) (left). The points in the training ($\mathbf{Y}$) are inflated and highlighted for visibility, but only the centre pixel and its class label are collected and used (middle). The predicted Heatmap of the model (right).

7.2.3 Data Augmentation

To improve the training of our network and examine its robustness to rotation, translation, scale, and noise, we apply spatial and pixel level augmentation to our training data for all models using Albumentations library [423]. In particular, we apply the following image transformations:

1. Randomly flip an image horizontally with a probability of 0.5.
2. Randomly flip an image vertically with a probability of 0.5.
3. Randomly shift and scale an image with shift limit of $0.0625°$, scale limit of $0.20°$ with a probability of 0.5.
4. Randomly rotate an image with a rotation limit of $20°$ with a probability of 0.5.
5. Randomly blur an image with blur limit of 1 with a probability of 0.3.
6. Randomly RGB-Shift an image with R-shift limit of 25, G-shift limit of 25, B-shift limit of 25 with a probability of 0.3. These augmentations help to further ensure robustness to shifts in lighting.

We did not apply any of the image transformation operations to our validation or test sets.

7.2.4 Performance Metrics

The following metrics were used to optimise and evaluate the model and to compare the quality of the predicted keypoint locations:

Euclidean distance measures the distance of the keypoints based on their coordinates (i.e the line segment between the two points), and does not depend on how the ground truth has been determined [424]. The best value of 0 indicates that the predicted keypoint is exactly at the same coordinate of the ground truth keypoint.

We calculate the sum of the squared Euclidean distance of the difference between two feature vectors, i.e., the predicted feature vector and the ground truth feature vector. This represents the total difference between the two feature vectors. The Euclidean distance is

$$d(g,p) = \sqrt{\sum_{i=1}^{n}(v_i^g - v_i^p)^2}, \tag{7.1}$$

where g and p are two sets of points in Euclidean n-space for ground truth and prediction, respectively. v_i^g, v_i^p are Euclidean vectors, starting from the origin of the space (initial point) for the ground truth and prediction, respectively. n is the number of keypoints.

Jensen-Shannon divergence is a distance measure between two distributions, such as the difference between the predicted and ground truth point distributions [425]. It can therefore be used to quantify the accuracy of the predicted keypoints. The lower this value is, the better the model performs.

This distance is calculated based on the Kullback-Leibler divergence (KLD) [426], where the inputs for the summation are probability distribution pairs. The KLD for two probability distributions, P and Q and when there are n pairs of predicted p, and ground truth g, can be expressed as:

$$KLD(P||Q) = \sum_{i=1}^{n} p_i(x) log\left(\frac{p_i(x)}{q_i(x)}\right), \tag{7.2}$$

to measure the difference between two probability distributions over the same variable x and indicate the dissimilarity between the distributions. The best value is 0. Utilising KLD, JSD can be expressed as follows:

$$JSD_M(P||Q) = \sqrt{\frac{KLD(p \parallel m) + KLD(q \parallel m)}{2}}, \tag{7.3}$$

where m is the point-wise mean of p and q.

This is a measure of the difference between two probability distributions P and Q. As can be seen from the formula, the best value of 0 indicates no difference between the distributions.

Object Keypoint Similarity (OKS) OKS keypoints estimation serves the same purpose as Intersection over Union (IoU) as in object detection. It is determined by dividing the distance between expected and ground truth points by the object's scale [427]. This gives the similarity between the keypoints (or corners) of the two detected boxes. The result is between 0 and 1, where 0 means no similarity between the keypoints, while perfect predictions will have OKS=1. The equation is as follows:

$$OKS = \frac{\sum_i \exp\left(-d_i^2/2s^2k_i^2\right)\delta\left(v_i > 0\right)}{\sum_i \delta\left(v_i > 0\right)}, \tag{7.4}$$

where d_i is the Euclidean distance between the detected keypoint and the corresponding ground truth, v_i is the visibility flag of the ground truth, s is the object scale, while $k_i s$ represents a per-keypoint constant that controls falloff.

To compute OKS, we pass the d_i through an unnormalized Gaussian with standard deviation $k_i s$. For each keypoint, this yields a keypoint similarity that ranges between 0 and 1. These similarities are averaged over all labelled keypoints. Given the OKS, we can compute Average Precision (AP) and Average Recall (AR) just as the IoU allows us to compute these metrics for box/segment detection.

Both equations 7.1 and 7.3 have been used for model training and optimisation, and also used to compare different models' performance as in table 7.1. Equation 7.4 was used as a final evaluation metric for all the models used in this study.

7.2.5 Model training

We trained six different models on the training subset. The models used for training are U-net [32], ResNet-18 [226], ShuffleNet-v2 [414], MobileNet-v2 [415], SqueezeNet [416] and our proposed lightweight network MFLD-net. For each experiment, we set our model hyperparameters to the same configuration for all models. All the models were trained with 224×224 resolution input and 56×56 heatmap resolution output except U-net [32] with 224×224 resolution for both input and output (see figure 7.6). Each model has two outputs (heatmap and coordinates), where two losses were applied as shown in figure 7.3.

We found that for this problem set, a learning rate of 1×10^{-3} works the best. It took around 50 epochs for all models to train on this problem and the learning rate was decayed by $\gamma = 0.1$ every 30 epochs. Our networks were trained on a Linux host with a single NVidia GeForce RTX 2080 Ti GPU using Pytorch framework [308]. The batch size we used was 64. We used Adam optimiser [204] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1.0 \times 10^{-08}$. We applied the same hyperparameter configuration for all six models.

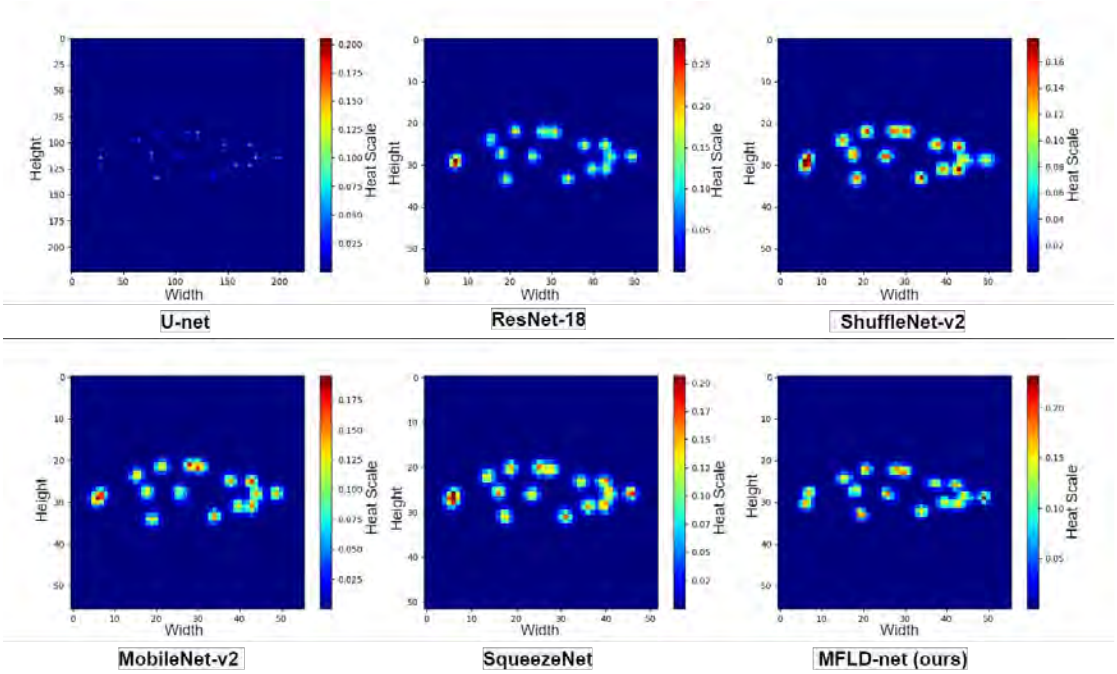

Figure 7.6: Sample output heatmap from each of the 6 networks used in this work.

The optimum model configuration will depend on the application, hence, these results are not intended to represent a complete search of model configurations.

Because we only used the training subset ($n = 1000$ images) for training and validation, during optimization, we heavily augmented our training set, challenging the model to learn a much broader data distribution than that in the training set. We applied several image transformations for data augmentation as specified in section 7.2.3.

We regarded the model to be converged when the validation loss stopped improving after 50 epochs. Only for the best performing version of the models, we calculated validation error as the Euclidean distance between predicted and ground-truth picture coordinates and Jensen-Shannon divergences between heatmaps and centres of the target Gaussians, which we assessed at the end of each epoch during optimization. Figure 7.7 shows the training and validation losses for our proposed network.

7.2.6 Model evaluation

Deep Learning (DL) models are typically evaluated for their predictive performance (i.e. ability to generalize to new data), using a sub-sample of annotated data (test set) that is not used for training or validation. A test set is typically used to avoid overfitting the model hyperparameters to the validation set, which can result in biased performance

measurements. Therefore, we used only 40% of our dataset for training and left the other 60% of its images for testing the model's predictive performance using metrics described in section 7.2.4.

Precision and Recall Object Keypoint Similarity (OKS) [427] was used as a performance metric (see section 7.2.4 for more details). As explained, the following 6 metrics are usually used for characterising the performance of a keypoint detector model. We, therefore, used them.

- Average Precision (AP):
  - AP (at $OKS = .50 : .05 : .95$ (primary metric))
  - $AP^{.50}$ (at $OKS = .50$)
  - $AP^{.75}$ (at $OKS = .75$)
- Average Recall (AR):
  - AR (at $OKS = .50 : .05 : .95$)
  - $AR^{.50}$ (at $OKS = .50$)
  - $AR^{.75}$ (at $OKS = .75$)

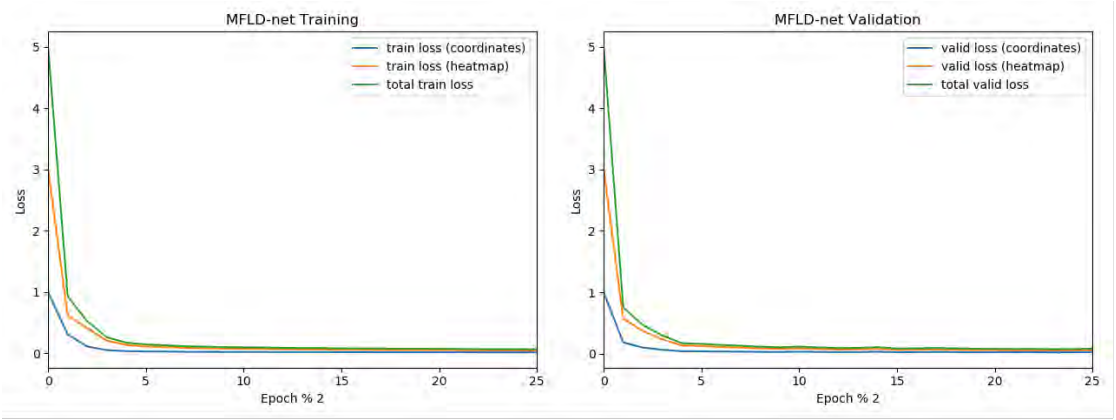

Figure 7.7: The two different losses, i.e. coordinate (Equ. 7.1) and heatmap (Equ. 7.3) prediction losses are shown along with the total loss for both training and validation.

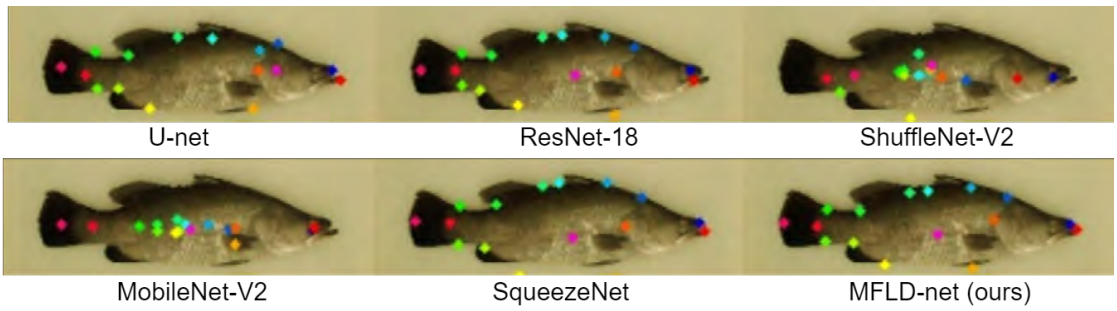

Figure 7.8: Example keypoints estimation predicted by the proposed network and a state of the art CNNs.

7.3 Results

To fully evaluate our model and compare it with other methods, we ran experiments to optimize our approach and compared it to the five aforementioned models in terms of image throughput (speed), accuracy, inference time, and generalization ability. We benchmarked these models using the test subset (see section 7.2.2 for details).

We applied the same training configuration for all of the six models, meaning that the models are all trained using the same dataset and data augmentations as explained in section 7.2.5.

7.3.1 Performance Comparison

Table 7.1 shows comparative results based on the number of parameters of a model, the model size on the hard disk, and the model throughput in image per second. In addition, the coordinates loss (Equ. 7.1), heatmap loss (Equ. 7.3), and the average of both losses are shown. All the tests were conducted on a desktop computer with a single NVidia GeForce RTX 2080 Ti GPU.

Overall, the results summarized in table 7.1 show that our network (MFLD-net) outperforms other networks, achieving the lowest number of parameters (47x fewer parameters than U-net [32]), the smallest size on the hard disk, and the second-highest throughput after SqueezeNet [416]. Also, our model has a lower average loss than U-net [32], ShuffleNet-v2 [414], and MobileNet-v2 [415]. The small number of parameters as well as the very compact size of our model while having a high throughput makes it an appealing solution for many problems such as real-time mobile fish video processing and portable autonomous systems [1].

Table 7.1: Performance Comparison to other models.

Network	# Params (x10^6)	Size (MB)	Throughput (img/sec)	Losses Coords [1]	Losses HeatMap [2]	Losses Avg. [3]
U-net [32]	31.04	124.3	201	0.024	0.355	0.190
ResNet-18 [226]	12.85	51.5	404	0.028	0.090	0.059
ShuffleNet-v2 [414]	3.06	12.5	170	0.047	0.153	0.100
MobileNet-v2 [415]	4.10	16.7	205	0.041	0.137	0.089
SqueezeNet [416]	2.33	9.4	551	0.027	0.078	0.052
MFLD-net (ours)	**0.65**	**2.7**	480	0.039	0.120	0.080

To examine the efficacy of our model generalisation, we compared its performance with randomly initialized weights, against the five benchmark models with randomly initialized weights to provide a direct comparison. We show in table 7.2 that our MFLD-net model achieves good generalisation with few training examples and without the use of transfer learning when combined with strong data augmentation. Overall, the results summarized in Table 7.2 show that our network (MFLD-net) outperforms ShuffleNet-v2 [414], MobileNet-v2 [415], and SqueezeNet [416] achieving $AP = 0.967$, while being competitive with U-Net and ResNet, despite having substantially fewer parameters. This shows the effectiveness and generalisability of our MFLD-net model.

Table 7.2: Performance comparison using the OKS metric on the **test** datasets.

Network	AP	$AP^{.50}$	$AP^{.75}$	AR	$AR^{.50}$	$AR^{.75}$
U-net [32]	0.968	0.990	0.990	0.983	0.999	0.999
ResNet-18 [226]	0.970	0.990	0.990	0.985	0.999	0.999
ShuffleNet-v2 [414]	0.949	0.990	0.990	0.968	0.999	0.999
MobileNet-v2 [415]	0.952	0.990	0.989	0.967	0.999	0.996
SqueezeNet [416]	0.964	0.990	0.990	0.975	0.999	0.999
MFLD-net (ours)	0.967	0.990	0.990	0.983	0.999	0.999

[1] This loss corresponds to the coordinates loss (Equ. 7.1).
[2] This loss corresponds to heatmap loss (Equ. 7.3).
[3] This loss corresponds to the average of both losses (Equ. 7.1 and Equ. 7.3).

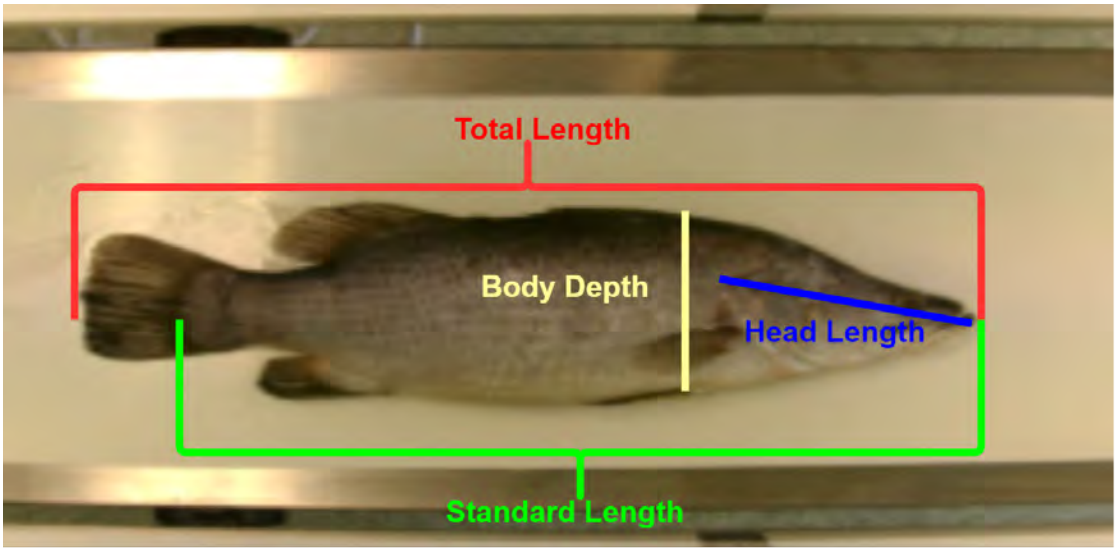

Figure 7.9: Fish body measurement used in this study.

7.3.2 Qualitative Results

To further confirm our model generalization on the unseen images, we perform a qualitative experiment on the test subset, with sample results shown in figure 7.8. This figure clearly shows that our network performs better than the previous methods. The other methods have the problem of misclassifying the pixels with a similar intensity of one colour as the other colour, whereas our method shows a strong ability to differentiate pixels with similar intensity. We can also clearly see that the proposed method can work on images with different lighting conditions.

7.4 Fish Morphometry

Morphometry is the study of the size and shape of organisms and their variation. Fish morphometry is a useful tool for fishery science, as it can help identify different species, populations, stocks, and growth patterns of fish [428]. Fish morphometry can be performed using traditional methods, such as measuring various body parts with a ruler or a caliper, or using advanced methods, such as image analysis, and deep learning. These methods provide an efficient approach to extract more information on the shape and variation of fish, automatically and cost-effectively.

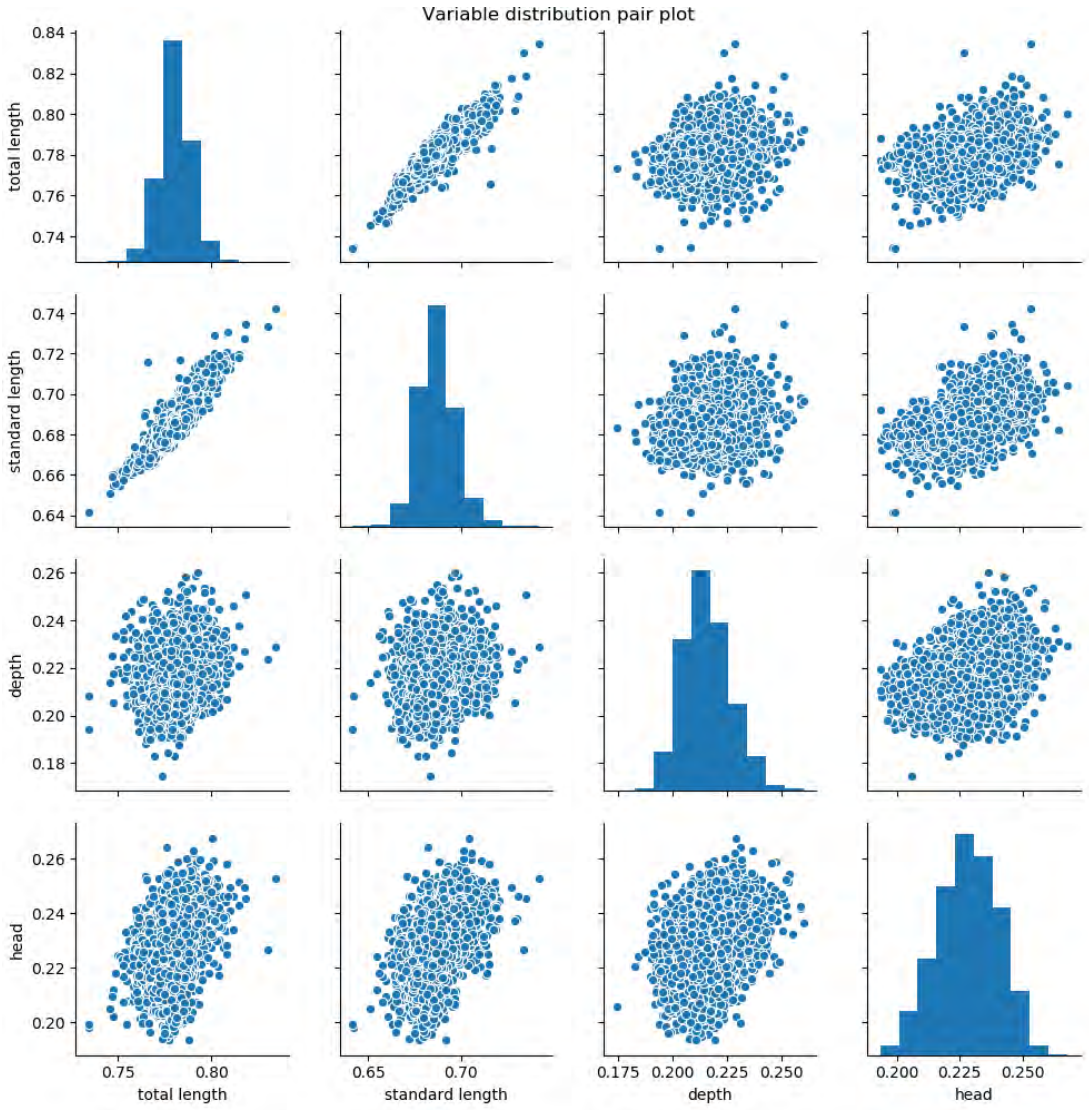

Figure 7.10: The distribution pair plots for the four body measurements (total length, standard length, body depth, and head length) used to describe the morphometry of fish.

Table 7.3: Performance Comparison of various DL models in measuring four important fish morphological traits. The mean absolute difference (MAD), and the standard deviation of the difference (SDD) between the manual and DL measurements in (mm) are shown. The Best Two Results are shown In RED and BLUE Colours.

Network	Total Length		Standard Length		Body Depth		Head Length	
	MAD	SDD	MAD	SDD	MAD	SDD	MAD	SDD
U-net [32]	10.57	10.67	08.00	08.64	06.23	06.04	06.58	06.77
ResNet-18 [226]	10.07	10.64	09.28	09.82	09.06	09.31	12.96	12.03
ShuffleNet-v2 [414]	14.11	14.95	12.31	12.05	11.90	11.21	15.48	15.70
MobileNet-v2 [415]	15.45	15.04	14.33	14.78	16.73	16.87	11.80	12.89
SqueezeNet [416]	10.07	10.04	09.28	09.64	09.06	09.20	12.96	11.38
MFLD-net (ours)	09.25	07.60	08.27	09.74	07.62	07.34	06.57	06.57

7.4.1 Fish body measurement used in this study

In this study, we used four body measurements to describe the morphometry of fish: total length, standard length, body depth, and head length. These important morphological measurements are widely used in monitoring fish, for example, its growth [429]. The four measurements automated using our approach are depicted in 7.9 and are defined as follows:

Total length is the overall length of the fish, measured from the tip of the snout to the end of the tail fin. This measurement is important for determining the overall size of the fish, which is relevant for various ecological and management purposes, such as estimating growth rates, biomass, and abundance.

Standard length is the length of the fish from the tip of the snout to the end of the vertebral column, excluding the caudal fin. This measurement is more appropriate for comparing the body proportions and shape of fish among different species or populations, as it removes the variation introduced by the size of the tail fin.

Body depth is the maximum vertical distance between the dorsal and ventral body surfaces, usually measured at the midpoint of the body length. This measurement reflects the thickness or robustness of the fish body, which can be related to its feeding habits, swimming ability, and reproductive strategy.

Head length is the distance from the tip of the snout to the posterior margin of the

operculum, which covers the gills. This measurement is relevant for assessing the size and shape of the head, which can provide information on the feeding behaviour, sensory perception, and phylogenetic relationships of the fish.

Therefore, the combination of these four measurements can provide a comprehensive description of the size, shape, and body structure of fish, which can be useful for various research and management applications. Figure 7.10 shows these measurements distribution pair plots based on the automatic measurements captured by MFLD-Net. These plots are essential to show the distribution and correlation of these measurements, which can provide insights into the fish's morphometry and body structure.

We are presenting these plots to demonstrate the effectiveness of our approach in automatically extracting these important morphological measurements from fish images. The automatic measurements captured by MFLD-Net can provide accurate and consistent results, which can save time and effort compared to manual methods.

7.4.2 Quantitative Comparison

In the quantitative comparison of fish morphometry, it is important to compare the accuracy and precision of the DL measurements. Here, accuracy refers to how close the DL measurements are to their true manual measurements, while precision refers to the degree of consistency or reproducibility of results.

To assess accuracy and precision, the following metrics have been used in this study: the mean absolute difference (MAD), and the standard deviation of the difference (SDD) between the manual and DL measurements.

MAD is the average absolute difference between two values (accuracy). It is calculated by taking the sum of the absolute differences between each value and dividing it by the number of values. The formula for calculating MAD is:

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|(x_i - y_i)|$$

where n is the total number of observations, x_i and y_i are the values of the $i-th$ observation in two different samples (here manual and DL measurements), and the vertical bars indicate absolute value.

SDD is a statistical measure that describes the amount of variation or dispersion between two sets of data. Specifically, it measures how spread out the differences between the two sets of data are (i.e. precision). SDD is calculated by taking the square root of the

variance of the differences using the following formula:

$$SDD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - y_i - \bar{x} + \bar{y})^2}{n-1}}$$

where x and y are the two sets of data, n is the number of observations in each set, and $\bar{x}$ and $\bar{y}$ are the means of the two sets.

Table 7.3 compares the performance of our MFLD-net model to other models in measuring fish morphometric traits such as total length, standard length, body depth, and head length. MAD and SDD between manual and deep learning (DL) measurements are reported in mm for each model. The top two results are highlighted in red and blue.

The MFLD-net model, proposed in this study, outperforms the others with MAD values of 9.25 and 6.57 for total length and head length. It also performs well for standard length and body depth with MAD values of 8.27 and 7.62, highlighted in blue. In addition, its SDD values are competitive with other models.

The U-net model shows the second-best performance and the ResNet-18 model performs well for standard length and body depth with MAD values of 9.28 and 9.06 but performs poorly for head length. The ShuffleNet-v2 and MobileNet-v2 models have higher MAD and SDD values for all traits. The SqueezeNet model performs well for total length but has high MAD values for standard length and head length.

7.5 Discussion

Fish morphology determination is required for both selecting and evaluating novel fish strains for cultivation. The most widely used method to characterise fish is by observation of their overall appearance. An experienced observer can determine a fish's size, weight, possibly sex, and even its condition. The traditional observation method to evaluate fish morphology includes weighing fish, measuring lengths with a ruler or calipers and or some other aspect of the fish, and then recording these observations. This observation process is slow, labour-intensive, and highly prone to human error.

A possible solution could automate the fish observation process if an accurate mobile system is developed that can be deployed in the field and in fish farms. This fish morphometric tool could quickly measure various fish features and morphological traits from fish images captured online or offline using a camera. The tool also collects the morphological data, and then uses it for analysis and producing a final report. Such a tool is very useful to aquaculture and fish farms and could provide a new way to select, evaluate, and

analyze fish and other aquaculture animal products.

In this paper, we developed a novel deep learning algorithm for accurate fish morphometric measurements from fish images. To efficiently measure various fish morphological traits, we developed a fish-specific landmark detection model that could accurately localise keypoints (landmarks) on the fish body (for example see figure 7.8). These landmarks can be then used to rapidly measure various fish traits including their weight, length, head shape, and body shape. In addition, the fish landmarks can be used to describe shape variation, deformation and differential development in various fish species [35, 429].

We build our land-mark detection model upon the most widely-used deep learning variant, i.e. CNN. A number of factors can significantly influence CNNs performance. These factors include the size of the network (including the number of layers, number of kernels, and their width), the number of input features, and the size of the training set. In addition, the use of the convolution layers affects the size and complexity of the network but can help to decrease the error rate and improve prediction accuracy. However, there is no clear mechanism to arrive at the optimal convolutional architecture for a specific task. The architecture selection involves choosing important hyperparameters such as the network structures and the training time.

Through experimentation and using our experience in developing deep learning algorithms, we designed a lightweight CNN with a short training time and high generalisability to make it suitable for fast deployment and real-time mobile applications in fish farms. Our experiments showed that MFLD-net best performances can be achieved by (i) increasing the size of the kernel to 9, (ii) including more input dimensions by Patch embeddings, and (iii) reducing the number of convolution layers to 8. The reduction in the number of convolution layers resulted in a model with fewer parameters that achieved better generalisation capabilities compared to the state-of-the-art models.

To train and evaluate our model performance, we collected a dataset containing 2500 harvested or sedated fish images. These images were manually annotated for important landmarks on the fish body. We used a combination of data augmentation techniques to improve the networks performance in a low data regime. In our experiments, the input images were scaled to a size of 224×224, and the output was the position of each fish landmark. These landmarks (keypoints) were indicated by a single, two-dimensional, symmetric Gaussian heatmap, where a scalar peak value reflects the prediction's confidence score. The quantitative and qualitative experimental results showed that our proposed model while being significantly lighter, can outperform some and be competitive

with other state-of-the-art models. We also showed that our model has a high generalisation capability and does not need transfer learning even when using a small training dataset.

The main limitation of our study is that all the samples for training and testing are taken from a similar source, even though, they were slightly different, due to being collected in different conditions and by different operators. Another limitation is the use of a single fish species in our dataset. Since there are a variety of different species and sizes of fish in the aquaculture industry, there is a need to test the model for more than one species. However, our aim in this study was to build a proof of concept, which can be extended in future works to other species. Our presented results indicate that our developed MFLD-net model trained using images from a single species could be generalised to detect fish of different species and in different environments. This could be the subject of future research.

Furthermore, we should emphasize that our model is not designed to classify fish species, but rather to detect landmarks on fish bodies that can be used for morphometric analysis. However, it is possible to extend our model to handle different fish species by using techniques such as multi-task learning or domain adaptation. For example, multi-task learning could be used to train our model to simultaneously detect landmarks and classify fish species, while domain adaptation techniques could be used to adapt our model to new fish species with minimal additional training data. These are potential avenues for future research and development of our model.

In addition, in future work, the model can be trained with images of other objects, or images captured from different fish species. It is worth noting that, collecting new fish images and annotating them is a time-consuming and expensive exercise. This was the case, even in our data collection trials, where fish images were collected when the fish passed on a conveyor belt and under a camera capturing videos.

While the proposed MFLD-net method has shown promising results in laboratory settings, there are several potential limitations and challenges that must be considered when deploying this method in real-world aquaculture farms. These include factors such as hardware requirements, scalability, and adaptability to different fish species.

In addition, developing new low-cost, low-power, and high-speed mobile devices has been an evolving research area in many applications such as agriculture [153], and marine science [174, 417]. These devices need a lightweight and fast network, such as the proposed model in this work. Therefore, an interesting future research project is to develop a low-cost mobile device to perform fish morphology estimation using the proposed

network.

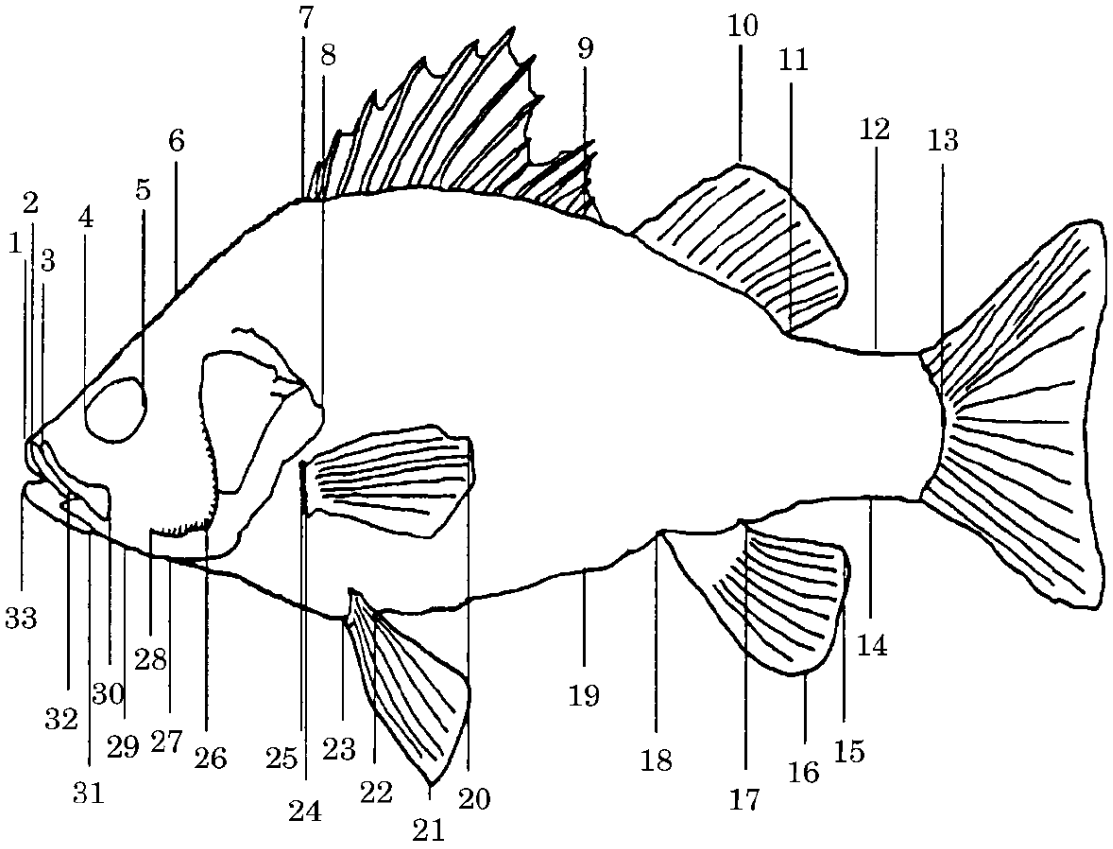

Figure 7.11: Position of most of the landmark points used to describe shape variation in *M. novemaculeata* from seven geographically distinct rivers. See [35] for an explanation of variables measured. The figure is from [35].

7.6 Conclusion

In conclusion, our research demonstrated the potential of using a vision transformer-inspired CNN for fish landmark detection and morphology measurement. Our proposed model outperforms existing deep learning models in terms of accuracy and speed while using fewer parameters, making it suitable for deployment on mobile and resource-constrained devices. This advancement brings us closer to practical applications in the rapidly growing aquaculture and fisheries industries. Future research will focus on testing the model on a wider range of aquaculture animals and exploring other CNN architectures for fish landmark detection.

Chapter 8

Prawn Morphometrics and Weight Estimation from Images using Deep Learning for Landmark Localization

Accurate weight estimation and morphometric analyses are useful in aquaculture for optimizing feeding, predicting harvest yields, identifying desirable traits for selective breeding, grading processes, and monitoring the health status of production animals. However, the collection of phenotypic data through traditional manual approaches at industrial scales and in real-time is time-consuming, labour-intensive, and prone to errors. Digital imaging of individuals and subsequent training of prediction models using Deep Learning (DL) has the potential to rapidly and accurately acquire phenotypic data from aquaculture species. In this Chapter, the third research question is addressed. Specifically, we applied a novel DL approach to automate weight estimation and morphometric analysis using the black tiger prawn (*Penaeus monodon*) as a model crustacean. The DL approach comprises two main components: a feature extraction module that efficiently combines low-level and high-level features using the Kronecker product operation; followed by a landmark localization module that then uses these features to predict the coordinates of key morphological points (landmarks) on the prawn body. Once these landmarks were extracted, weight was estimated using a weight regression module based on the extracted landmarks using a fully connected network. For morphometric analyses, we utilized the detected landmarks to derive five important prawn traits. Principal Component Analysis (PCA) was also used to identify landmark-derived distances, which were found to be highly correlated with shape features such as body length, and width. We evaluated our approach on a large dataset of 8164 images of the Black tiger prawn (Penaeus monodon) collected from Australian farms. Our experimental results demonstrate that the novel DL approach

outperforms existing DL methods in terms of accuracy, robustness, and efficiency.

8.1 Introduction

The farming of marine prawns (shrimp) is one of the most important and largest aquaculture production sectors globally [430]. As in any animal production sector, the acquisition of industrial-scale data on weight and other commercially relevant phenotypic traits is important, as this data can improve yields and economic efficiency [431] through informing pond management, feeding, grading and selective breeding processes [432]. However, the current collection of this data is manual using traditional weight and morphometric analyses that are often invasive, time-consuming, labour-intensive, and prone to human error. Therefore, there is a need for the development of automated, fast, and accurate methods for weight estimation and associated morphometric analyses.

Computer vision and image analysis enabled by Deep Learning (DL) have emerged as promising techniques for solving various problems in the Internet of Underwater Things (IoUT) [174] and equally in aquaculture [1,321]. In particular, image analysis can be used to identify prawn species, detect prawns in images, measure prawn length, and estimate prawn weight [433]. However, existing methods have some limitations, such as requiring high-quality images with uniform backgrounds, relying on hand-crafted features that may not capture complex variations, or using simple regression models that may not generalize well to different conditions [434]. Moreover, most of these methods do not consider the morphological characteristics of prawns that affect their weight distribution.

In this paper, we propose a novel Deep Learning [9, 17] approach for automated morphometric analyses and weight estimation of prawns from digital images. Our approach consists of two main components: a Kronecker product-based feature extraction module (KPFEM), and a landmark localization module (LLM). The KPFEM uses the Kronecker product operation [435] to combine low-level and high-level features from different convolutional layers efficiently. The LLM predicts the coordinates of key points on the prawn body using a localization network. For weight estimations, we have designed a weight regression module (WRM) that works based on the extracted landmarks using a fully connected network. We also use the landmarks generated by the LLM component to perform morphometric analysis. To the best of our knowledge, this is the first work that applies the Kronecker product operation for feature extraction in morphometrics analysis. Moreover, this is the first work that uses a localization network for landmark detection in prawn images.

The main contributions of our paper are as follows:

1. We introduce a novel feature extraction method based on Kronecker product that can capture rich semantic information from different scales from the prawn image while offering advantages such as reduced model parameters that make it well-suited for resource-constrained devices in aquaculture settings, and improved performance compared to traditional CNNs.

2. We apply a deep network for landmark localization that can handle occlusions and deformations better than existing methods. This is essential in scenarios such as bulk aquaculture product monitoring.

3. We design a weight regression model that incorporates morphometric features derived from landmarks to improve weight prediction accuracy.

4. We perform morphometric shape analyses on five important prawn traits derived from landmark data, and also use Principal Component Analysis (PCA) to find landmarks correlated with shape features.

5. We evaluate our approach on a large dataset of prawn images and show that it outperforms existing methods in accuracy, robustness, and efficiency.

8.2 Method

Our proposed deep learning architecture is shown in Fig. 8.1 with two distinct outputs, one for prawn weight, and the second for landmark identification and applied morphometric analyses.

The KPFEM serves as a feature extraction module and utilizes Kronecker convolution operation in a novel network architecture to extract features from the input prawn image. The resulting feature map is then passed to the LLM, which uses a deep learning-based approach to detect 12 landmarks on the prawn body.

The predicted landmarks are then used to calculate the distances between any 12 landmarks, resulting in a total of $(12(12-1)/2 = 66)$ possible distances. These distances are then fed to the WRM, which uses a deep learning-based approach to predict the weight of the prawn. The 12 detected landmarks are also used to perform morphometric analysis of the prawn. The following subsections provide further details on the role and importance of each of the aforementioned modules in reaching the overall goal of our architecture, i.e. automatic weight estimation and morphometric analysis from prawn images.

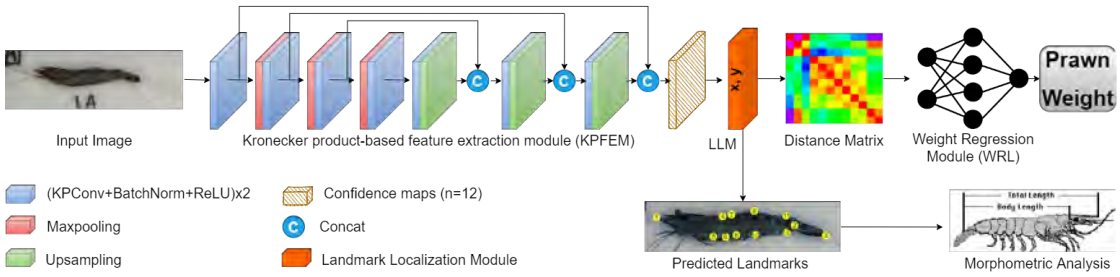

Figure 8.1: An overview of our proposed network architecture used for landmark detection from images that can be used for weight estimation and morphometric analyses. Our architecture consists of two main modules: a Kronecker product-based feature extraction module (KPFEM) and a subsequent landmark localization module (LLM). These are followed by a weight regression module (WRM). The KPFEM module is responsible for extracting features from the input image using Kronecker product-based convolutional layers, while the LLM module localizes the landmarks of the prawn using the extracted features. The WRM module regresses the weight of the prawn based on the detected landmarks. This multi-stage approach has shown promising results for accurate prawn weight and morphometric estimation from images.

8.2.1 Kronecker product-based feature extraction module (KPFEM)

The first module in our architecture is based upon the Kronecker convolution operation implemented within several Kronecker Convolution Layers (KCL), which are explained in detail below. We also provide an explanation of the advantages that KCL provides for our architecture, compared to conventional CNNs.

Kronecker Convolution Layer

One of the main strengths of this work, which makes it suitable for developing a reliable tool for the challenging task of prawn image analysis, is leveraging the Kronecker convolution operation in a unique way to extract more informative features from the input image, resulting in more accurate landmark detection and weight estimation.

The Kronecker convolution operation is based on the Kronecker product which is a mathematical operation that takes two matrices and produces a new matrix that is formed by multiplying each element of the first matrix with the second matrix. In the context of convolutional neural networks, the Kronecker product is used to create a weight tensor that is a Kronecker product of two smaller tensors, one that represents the filters in the convolutional layer and another that represents the input image, to create a large weight

tensor. This weight tensor is then used to compute the convolution between the input image and the filter.

In contrast to standard convolutional layers, which apply a single filter across all input channels to find local spatial relationships between input features, the Kronecker convolution operation applies a filter that captures both spatial and higher-order information across all input channels. This makes it an effective approach for feature extraction in multi-channel inputs such as RGB images.

The Kronecker product can be written as follows:

$$\underset{(n\times n)}{[\mathbf{A}_1]} \otimes \underset{\left(\frac{s}{n}\times\frac{d}{n}\times k\times k\right)}{\begin{bmatrix} \mathbf{F}_1 \end{bmatrix}} + \ldots + \underset{(n\times n)}{[\mathbf{A}_n]} \otimes \underset{\left(\frac{s}{n}\times\frac{d}{n}\times k\times k\right)}{\begin{bmatrix} \mathbf{F}_n \end{bmatrix}} = \underset{(s\times d\times k\times k)}{\begin{bmatrix} \mathbf{H} \end{bmatrix}}. \tag{8.1}$$

where matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and the filter matrix $\mathbf{F}$ and the weight matrix $\mathbf{H}$ have the same dimensions: s channels, d filters, and $k \times k$ kernel size.

The Kronecker Convolution Layer (KCL) uses the Kronecker product to arrange convolution filters in a way that reduces the number of parameters by a factor of $1/n$. We explain how this works for different values of n in Eq. 8.1. When $n = 1$, we have a real-valued convolution and the Kronecker product is just a scalar multiplication. The filter matrix $\mathbf{F}$ has the same size as the weight matrix $\mathbf{H}$, which is $s \times d \times k \times k$.

When $n = 2$, we have a complex-valued convolution and the Kronecker product is between two matrices. The filter matrices $\mathbf{F}_1$ and $\mathbf{F}_2$ are half the size of $\mathbf{H}$, and they contain the filters for each complex component. The algebra is done with matrices $\mathbf{A}_1$ and $\mathbf{A}_2$. This way, we use half as many parameters as in the real case. When $n > 2$, we can extend this idea by using smaller filter matrices for each dimension. The size of $\mathbf{H}$ does not change, but the parameter size *decreases* with higher values of n.

The weight tensor $\mathbf{H}$ in the KCL layer is obtained by summing Kronecker products between two groups of learnable matrices. Specifically, it can be expressed as:

$$\mathbf{H} = \sum_{i=1}^{n} \mathbf{A}_i \otimes \mathbf{F}_i, \tag{8.2}$$

where $\mathbf{A}_i$ is a $n \times n$ matrix that describes the algebra rules, and $\mathbf{F}_i$ is a $\frac{s}{n} \times \frac{d}{n} \times k \times k$ matrix representing the i-th batch of filters. These filters are arranged according to the algebra rules to construct the final weight matrix.

Algorithm 4 is a pseudocode implementation of Kronecker Product Convolution using PyTorch-like syntax. The algorithm takes two input tensors: A, a 2D tensor of shape (height, width), and F, a 4D tensor of shape ($num\_filters$, $num\_channels$, $filter\_height$, $filter\_width$), and performs Kronecker product convolution on them. The output tensor has a shape of ($num\_filters$, $num\_channels$, $output\_height$, $output\_width$), where $output\_height$ and $output\_width$ are calculated based on the size of A and the filter size.

The algorithm works by first computing the Kronecker product of A and F, which is a block matrix of shape ($num\_filters$, $num\_channels$, $filter\_height$, $filter\_width$). This is achieved by expanding the dimensions of A and F and multiplying them element-wise. The resulting block matrix is then reshaped to the desired output shape.

Algorithm 4: Kronecker Product, PyTorch-like

```
import torch

def kronecker_product(self, A, F):
    mtx1 = torch.Size("torch.tensor"(A.shape[-2:]) * "torch.tensor"(F.
    shape[-4:-2]))
    mtx2 = torch.Size("torch.tensor"(F.shape[-2:]))
    res = A.unsqueeze(-1).unsqueeze(-3).unsqueeze(-1).unsqueeze(-1) *
    F.unsqueeze(-4).unsqueeze(-6)
    mtx0 = res.shape[:1]
    out = res.reshape(mtx0 + mtx1 + mtx2)
    return out
```

Standard Convolutional Layer

A standard convolutional layer convolves the input $\mathbf{x} \in \mathbb{R}^{t \times s}$ with the filter tensor $\mathbf{W} \in \mathbb{R}^{s \times d \times k \times k}$ to generate the output $\mathbf{y} \in \mathbb{R}^{d \times t}$, as follows:

$$\mathbf{y} = \text{Conv}(\mathbf{x}) = \mathbf{W} * \mathbf{x} + \mathbf{b}, \tag{8.3}$$

where s is the input channels dimension, d the output, k is the filter size, and t is the input and output dimension. The bias term $\mathbf{b}$ has negligible impact on the number of parameters, resulting in a complexity of $\mathcal{O}(sdk^2)$.

The KCL layer is a convolutional layer that uses a weight tensor $\mathbf{H}$ to organize its

filters, which is constructed by summing Kronecker products. The layer can be defined as

$$\mathbf{y} = \text{KCL}(\mathbf{x}) = \mathbf{H} * \mathbf{x} + \mathbf{b}, \tag{8.4}$$

where $\mathbf{H}$ is a learnable tensor with dimensions $s \times d \times k \times k$. The two groups of learnable matrices used to construct $\mathbf{H}$ are denoted as $\mathbf{A}_n$ and $\mathbf{F}_n$, which are combined through Kronecker products to create $\mathbf{H}$, (see Eq. (8.1) and Eq. (8.2)). The value of n can be set by the user to specify the real or hypercomplex domain, and controls the degree of freedom of $\mathbf{A}_n$ and $\mathbf{F}_n$. The number of parameters in the KCL layer is reduced by a factor of $1/n$ compared to a standard convolutional layer in real-world problems, because typically $sdk^2 \gg n^3$. During training, the matrices $\mathbf{A}_n$ and $\mathbf{F}_n$ are learned and used to construct $\mathbf{H}$. The dimensions of $\mathbf{F}_n$ are $\frac{s}{n} \times \frac{d}{n} \times k \times k$ for squared kernels, and $\frac{s}{n} \times \frac{d}{n} \times k$ for 1D kernels. Hence, The KLC complexity of the weight matrix can be approximated to $\mathcal{O}(sdk^2/n)$.

KCL advantages compared to standard convolution

Compared to standard convolutional layers, using KCL brings several advantages. As discussed, firstly, the KCL layer reduces the number of parameters by a factor of $1/n$ in real-world problems, where n is the hyperparameter that specifies the desired domain. For example, for RGB images that have $n = 3$, the network number of parameters is reduced by 66%. This reduction in parameters can lead to faster training and inference times, as well as reduced memory usage, making KCL well-suited for resource-constrained devices. Secondly, KCL allows for weight sharing among different channels in multidimensional data, such as colour images, which enables capturing latent intra-channel relations that standard convolutional networks may ignore due to the fixed structure of the weights. This can result in better performance in tasks that involve correlated channels. Finally, the KCL layer can be easily integrated into any convolutional model by replacing standard convolution or transposed convolution operations, and the hyperparameter n provides high flexibility to adapt the layer to any kind of input. Overall, the KCL layer offers a promising alternative to standard convolutional layers and has the potential to improve the performance of convolutional neural networks in various applications.

Feature Extraction Structure

The proposed KPFEM module is composed of 14 KCLs that extract features from the input prawn image. As shown in Fig. 8.1 each layer is followed by a rectified linear unit

(ReLU) activation function and every second layer is followed by a max pooling operation to reduce the spatial dimension of the feature maps. The module has ”skip connections”, which allow information to flow through the network more efficiently by skipping over certain layers that might otherwise impede the flow of information.

8.2.2 Landmark Localization Module (LLM)

The landmark localization module takes the extracted features generated by the KPFEM module and predicts the locations of the landmarks. Therefore, for our keypoint detection task, instead of using FCN to directly predict a numerical value of each keypoint coordinate as an output (i.e. regressing images to coordinate values), we modified FCN to predict a stack of output heatmaps (i.e. confidence maps), one for each keypoint. The position of each keypoint is indicated by a single, two-dimensional, symmetric Gaussian in each heatmap in the output, and the scalar value of the peak reflects the prediction's confidence score.

Moreover, our proposed LLM not only predicts heatmaps but also predicts scalar values for coordinates of each keypoint. Therefore, during the training process, we have a multi-task loss function, which consists of two losses, i.e. JensenShannon divergence for heatmaps and Euclidean distance for coordinates. The first loss measures the distances between the predicted heatmaps and the ground-truth heatmaps, while the second loss measures the distances between the predicted coordinates and the ground-truth coordinates. Then, we take the average of the two losses as the optimization loss.

As demonstrated in Fig. 8.1, the LLM output is a set of predicted landmark coordinates, using which also a distance matrix is produced to feed to the next module in our proposed architecture.

8.2.3 Weight Regression Module (WRM)

The final component of our architecture is the weight regression module which is made up of a multilayer perceptron (MLP) that consists of five layers of nodes: an input layer, three hidden layers, and an output layer. Each node in the hidden layers applies the ReLU activation function to the weighted sum of inputs from the previous layer.

The weight regression module takes the output of the landmark localization module, which includes the predicted locations of the landmarks, and measures 66 distances between any 12 landmarks to predict the weight. We use the distances between the landmarks to estimate significant traits such as total length, body length, carapace length, and

length-width ratios. These measurements are typically made between easily distinguishable landmarks. The distances between them are assigned a trait name and then used as inputs for a pre-trained regression model that maps these estimates to the prawn weight. Specifically, we generated morphological measurements using a combination of methods from previous studies and those that were possible within the constraints of the available images. For the morphometric shape analysis, we used 8,164 photographed specimens to estimate 66 morphometric distances derived from 12 landmarks. These distances were then used by the weight regression module to predict the prawn weight directly from the landmarks. This is a practical and useful application of our model. We trained our model using a mean squared error (MSE) loss to minimize the difference between the predicted weight and the ground-truth weight. This ensures that our model is accurate and reliable for predicting prawn weights based on landmark information.

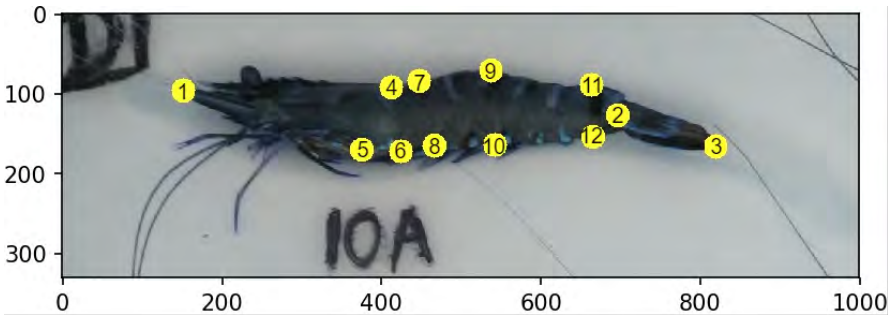

Figure 8.2: Keypoints (landmarks) of interest marked on the body of The Black Tiger Prawn (*Penaeus monodon*)

8.3 Experiments

In this section, we present the experimental setup and results of our proposed approach for automated morphometric analysis and weight estimation of prawns from images. We first describe the dataset used for the experiments and the data preprocessing steps. Next, we provide details on the training and evaluation of our approach and compare it to existing methods. Finally, we analyze the results and discuss the strengths and limitations of our approach.

8.3.1 The Dataset

We collected images of 8164 individual Black Tiger Prawns (*Penaeus monodon*) from an aquaculture farm in Australia. The images were captured using a digital camera under

natural lighting and saved as JPG files. We used ImageJ software to annotate each image with 12 landmarks that correspond to prawn size and shape (Fig. 8.2). The landmark positions were defined based on a previous study [436] and adapted to our image quality.

To evaluate the reliability of the landmark data, we measured 300 images twice by one operator (O1) and once by another operator (O2) and computed the intraclass correlation coefficient (ICC) for each landmark pair using R software. The ICC values ranged from 0.96 to 0.99, indicating high agreement between operators. The annotated images contain 12 specific and homologous points on the prawn body, including eyes, antennae, and tail. For a complete list of the 12 landmarks, see Table 8.1. We then derived 5 traits from these landmarks, which are listed in Table 8.2 and used for morphometric analyses in Sec. 8.4.4.

The dataset, therefore, includes for each image, the 12 ground-truth landmark coordinates, and morphometric measurements obtained from the annotated coordinates on images. The dataset was divided into three parts: training, validation, and testing. We used the training dataset to train the Deep Learning models, the validation dataset to assess model performance during training, and the testing dataset to evaluate the performance of the trained models. The ground truth landmarks for each prawn image had the form $[(x_1, y_1), .., (x_k, y_k)]$, where (x_i, y_i) represented the ith landmark location. Our model was then trained to predict keypoint locations for each prawn image as shown in Fig. 8.2. The predicted landmark had the same form as the ground truth.

Using a well-curated and annotated dataset is critical for the success of Deep Learning-based morphometric analysis. It provides the models with sufficient training data to learn shape information from the images and accurately identify landmark points. To achieve this, we carefully curated the dataset to ensure a balanced representation of different prawn species and body shapes, to enhance the robustness of the results. We also pre-processed the images to make sure they were of similar size and resolution and to eliminate background noise and other distractions that could impact the accuracy of landmark identification.

8.3.2 Evaluation Metrics

In this section, we describe the performance metrics used to optimize and evaluate the model and compare the quality of the predicted keypoint locations.

Table 8.1: Lists of the 12 landmarks used in our analysis and their descriptions

Landmark number	**Landmark Description**
1	Placed on the most anterior point of the antennal scale
2	Placed on the most anterior point of the tail
3	Placed on the most posterior point of the tail
4	Placed on the junction of the carapace and abdomen placed at the most dorsal point
5	Placed on the midway along the carapace on the ventral side of the prawn
6	Placed on the junction of the carapace and abdomen placed at the most ventral point
7	Placed dorsally on the midpoint of the first abdominal segment
8	Placed ventrally on the midpoint of the first abdominal segment
9	Placed dorsally on the midpoint of the third abdominal segment
10	Placed ventrally on the midpoint of the third abdominal segment
11	Placed dorsally on the midpoint of the last abdominal segment
12	Placed ventrally on the midpoint of the last abdominal segment

Euclidean distance

The first metric is the Euclidean distance, which measures the distance of the keypoints based on their coordinates and does not depend on how the ground truth has been determined. A value of 0 indicates that the predicted keypoint is exactly at the same coordinate as the ground truth keypoint. We calculate the sum of the squared Euclidean distance of the difference between two kyepointsi.e., the predicted keypoint and the ground truth human-annotated keypoint This represents the total difference between the two points. The equation for Euclidean distance is shown in Equation 8.5,

$$d(g,p) = \sqrt{\sum_{i=1}^{n}(v_i^g - v_i^p)^2}, \tag{8.5}$$

where g and p are two sets of points in Euclidean n-space for ground truth and prediction, respectively, v_i^g, v_i^p are Euclidean vectors starting from the origin of the space (initial point) for the ground truth and prediction, respectively, and n is the number of keypoints.

Table 8.2: Important prawn traits, their descriptions, and their corresponding landmark coordinates from Table 8.1

Trait	Landmark Coordinates	Trait Description
Total length	1-3	From the most anterior point of the antennal scale to the most posterior point of the tail
Body length	1-2	From the most anterior point of the antennal scale to the most anterior point of the tail
First abdominal segment height	7-8	From the dorsal midpoint to the ventral midpoint on the first abdominal segment
Third abdominal segment height	9-10	From the dorsal midpoint to the ventral midpoint on the third abdominal segment
Last abdominal segment height	11-12	From the dorsal midpoint to the ventral midpoint on the last abdominal segment

Jensen-Shannon divergence

The second metric is the Jensen-Shannon divergence, which is a distance measure between two distributions and can be used to quantify the accuracy of the predicted keypoints distribution compared to the ground-truth keypoints distribution. The lower the value, the better the model performs. This distance is calculated based on the Kullback-Leibler divergence (KLD) and can be expressed as shown in Equation 8.6.

$$KLD(P||Q) = \sum_{i=1}^{n} p_i(x) log\left(\frac{p_i(x)}{q_i(x)}\right), \tag{8.6}$$

where the KLD for two probability distributions P and Q, and when there are n pairs of predicted p, and ground truth q.

The Jensen-Shannon divergence (JSD) is then calculated using Equation 8.7,

$$JSD_M(P||Q) = \sqrt{\frac{KLD(p \parallel m) + KLD(q \parallel m)}{2}}, \tag{8.7}$$

where m is the point-wise mean of p and q. The JSD measures the difference between two probability distributions, with a value of 0 indicating no difference between the distributions.

Object Keypoint Similarity (OKS)

The third metric use to evaluate the performance of our landmark detection model is the Object Keypoint Similarity (OKS), which is determined by dividing the distance between expected and ground truth points by the object's scale. OKS keypoints estimation serves the same purpose as Intersection over Union (IoU) as in object detection. This gives the similarity between the keypoints of the two detected boxes, with a result between 0 and 1, where 0 means no similarity between the keypoints, while perfect predictions will have OKS=1. The equation for OKS is shown in Equation 8.8.

$$OKS = \frac{\sum_i \exp\left(-d_i^2/2s^2k_i^2\right)\delta\left(v_i > 0\right)}{\sum_i \delta\left(v_i > 0\right)}, \tag{8.8}$$

where d_i is the Euclidean distance between the detected keypoint and the corresponding ground truth, v_i is the visibility flag of the ground truth, s is the object scale, while k_i represents a per-keypoint constant that controls falloff.

Equations 8.5 and 8.7 were used for model training and optimization and also used to compare different models' performance, as shown in Table 8.3. Equation 8.8 was used as a final evaluation metric for all the models used in this study, as shown in Table 8.4.

Precision and Recall

Object Keypoint Similarity (OKS) [427] was used as a performance metric (see section 8.3.2 for more details). As explained, the following 6 metrics are usually used for characterising the performance of a keypoint detector model and were applied in our study.

- Average Precision (AP):
  - AP (at $OKS = .50 : .05 : .95$ (primary metric))
  - $AP^{.50}$ (at $OKS = .50$)
  - $AP^{.75}$ (at $OKS = .75$)
- Average Recall (AR):
  - AR (at $OKS = .50 : .05 : .95$)
  - $AR^{.50}$ (at $OKS = .50$)
  - $AR^{.75}$ (at $OKS = .75$)

8.3.3 Model training

In this section, we will describe the process of training our Deep Learning model to identify landmarks in prawn images. Our model architecture, which was described in Section 8.2, was trained using the aforementioned large dataset of annotated prawn images. The training process consisted of several key steps, including data preprocessing, model selection, and hyperparameter tuning.

Data Preprocessing

Before the training process can begin, the image data must be preprocessed to ensure that it is suitable for input into the Deep Learning model. This includes resizing the original image size of 1000×331, to a consistent size of 320×320, normalizing the pixel values, and converting the images to grayscale if necessary. In addition, the annotated landmark locations must be converted into a format that can be used by the model, such as a heatmap or a set of points. We applied some image transformation operations to our training set with certain probabilities. These operations are: Horizontal flip: 0.5, Vertical flip: 0.5, Shift and scale: 0.5 (shift limit = $0.0625°$, scale limit = $0.20°$), Rotation: 0.5 (rotation limit = $20°$), Blur: 0.3 (blur limit = 1), RGB-shift: 0.3 (R-shift limit = 25, G-shift limit = 25, B-shift limit = 25). These operations help to improve the robustness of our model to lighting changes. We did not transform the images in our validation or test sets in any way.

The ground truth landmarks for each prawn image are represented in the form of $[(x_1, y_1), .., (x_k, y_k)]$, where (x_i, y_i) denotes the location of the ith landmark. In the training process, both the original (x_i, y_i) values and the converted heatmap derived from these values are utilized in the loss function.

Model Selection

Once the data has been preprocessed, the next step is to select a model architecture that is suitable for our problem. There are a variety of Deep Learning models that can be used for image landmark identification. In this work, we have selected six models for our experiments: U-net [32], ResNet-18 [226], ShuffleNet-v2 [414], MobileNet-v2 [415], SqueezeNet [416], and our proposed KPFEM.

Hyperparameter Tuning

Once the model architecture has been selected, the next step is to tune the hyperparameters to achieve optimal performance. This includes selecting the optimal batch size, learning rate, and number of epochs. The hyperparameters are selected using a combination of grid search and cross-validation to ensure that the model is generalizing well and not overfitting to the training data. For this problem set, we chose a learning rate of 1×10^{-3} as the best option. All models took about 200 epochs to train on this problem and we reduced the learning rate by $\gamma = 0.001$ after every 50 epochs. We also used Adam optimiser [204] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1.0 \times 10^{-08}$. We applied these hyperparameters to all six models. The best model configuration may vary depending on the application, so these results do not cover all possible model configurations.

We split the dataset into three sets: "Train", "Validation", and "Test", comprising 40%, 20%, and 40% of the data, respectively. We trained all models on the Train subset of the data with the same hyperparameters. All models had two outputs (heatmap and coordinates) with two losses (see Sec. 8.3.2). All models took 320×320 input images and produced 56×56 output heatmaps except U-net which had 320×320 output images.

Training

The final step in the training process is to train the model using the preprocessed data and the optimized hyperparameters. The model is trained using a supervised learning approach, where the ground truth landmark locations are used to calculate the loss function and update the model parameters. The training process is repeated until the model has reached convergence or a maximum number of epochs has been reached. We used Pytorch framework [308] on a Linux host with a single NVidia GeForce RTX 2080 Ti GPU and a batch size of 64.

Once the model was trained, it was evaluated on the test subset of the dataset to assess its performance in identifying landmarks in new, unseen images. The evaluation metrics, described in the evaluation metrics section (Sec. 8.3.2), are used to quantify the accuracy of the model and provide insight into its strengths and weaknesses.

8.4 Results

In this section, we present the results of our experiments on prawn landmark detection, used for weight estimation and morphometric analyses from images using our proposed

approach. We conducted experiments to optimize our method and assess its performance against the five other models that we mentioned earlier, based on image throughput (speed), accuracy, inference time, and generalization ability. The test subset was used to measure these models (see section 8.3.1 for more information).

8.4.1 Landmark Detection

Table 8.3 presents a comparative analysis based on several metrics, including the number of floating-point operations (FLOPs), the number of parameters, model size, and throughput in images per second, as well as the coordinates loss (Equ. 8.5), heatmap loss (Equ. 8.7), and the average of both losses. It is worth noting that, the actual throughput of a network is the number of instances it can process in one second with the optimal batch size. It is obtained by dividing the total number of instances processed by the total time taken to process them and can vary depending on various factors such as the model's complexity, the input data size, and the available hardware resources.

Table 8.3: Landmark Detection Performance Comparison of our model compared to five benchmark models.

Network	FLOPs (x10^6)	#Params (x10^6)	Size (MB)	Throughput (img/sec)	Losses Coords	Losses HeatMap	Losses Avg.
U-net [32]	16.52	31.04	124.3	201	0.024	0.355	0.190
ResNet-18 [226]	2.62	12.85	51.5	404	0.028	0.090	0.059
ShuffleNet-v2 [414]	0.44	3.06	12.5	170	0.047	0.153	0.100
MobileNet-v2 [415]	0.67	4.10	16.7	205	0.041	0.137	0.089
SqueezeNet [416]	0.92	2.33	9.4	551	0.027	0.078	0.052
KPFEM (ours)	**0.01**	**0.39**	**1.6**	**562**	0.023	0.084	0.0053

The experimental results shown in Table 8.3, were conducted on a desktop computer equipped with a single NVidia GeForce RTX 2080 Ti GPU. These results demonstrate that our proposed network which is based on our KPFEM feature extraction method outperforms other networks in several aspects, including having the lowest number of parameters (47 times fewer parameters than U-net [32]), the smallest size on the hard disk, and the second-highest throughput, following SqueezeNet [416]. Moreover, our model exhibits a lower average loss than other popular models, including U-net [32], ShuffleNet-v2 [414], and MobileNet-v2 [415].

The low number of parameters, small model size, and high throughput of our model make it a promising solution for many real-time applications, such as mobile prawn video

processing and portable phenotyping systems [1]. However, there are some limitations that must be taken into consideration when interpreting the results. First, our experiments were conducted on a single GPU, which may not represent the complete profile performance, and may vary on other hardware configurations. Second, the dataset used for the experiments was limited in size and scope, and therefore, further studies on more diverse datasets are necessary to validate the effectiveness and generalizability of our approach. Future work will focus on addressing these limitations and further improving the efficiency and accuracy of our approach.

Table 8.4 shows the performance of our model on the test subset of the dataset compared to benchmark models in landmark detections, using the OKS evaluation metric. To assess the generalization effectiveness of our model, we compared the performance of our model with randomly initialized weights, against the other models with randomly initialized weights as well to provide a direct comparison. Table 8.4 demonstrates that our proposed model performs well in generalization with only 40% of the data and without the use of transfer learning, when combined with robust data augmentation techniques.

The overall outcome depicted in Table 8.4 indicates that our proposed network surpasses both ShuffleNet-v2 [414] and MobileNet-v2 [415] in terms of landmark detection performance, with an accuracy of $AP = 0.986$, while still competing with SqueezeNet [416] even though it has substantially fewer parameters. It is noteworthy that our model attains this high accuracy with only 0.39M parameters and without relying on transfer learning. These results clearly demonstrate the effectiveness and generalisability of our KPFEM-based model.

Table 8.4: Performance comparison using the OKS metric on the **test** dataset.

Network	AP	$AP^{.50}$	$AP^{.75}$	AR	$AR^{.50}$	$AR^{.75}$
U-net [32]	0.981	0.990	0.990	0.974	0.999	0.999
ResNet-18 [226]	0.984	0.990	0.990	0.982	0.999	0.999
ShuffleNet-v2 [414]	0.957	0.990	0.990	0.979	0.999	0.999
MobileNet-v2 [415]	0.963	0.990	0.989	0.979	0.999	0.996
SqueezeNet [416]	0.971	0.990	0.990	0.983	0.999	0.999
KPFEM (ours)	0.986	0.990	0.990	0.985	0.999	0.999

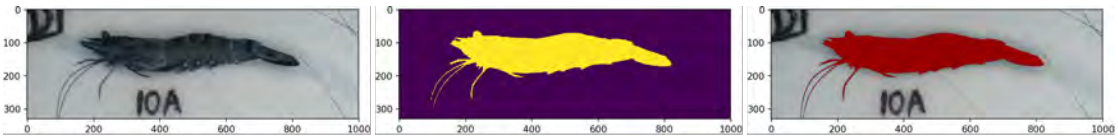

Figure 8.3: Example of threshold segmentation. From left to right: (a) original image, (b) segmented prawn body, (c) overlay of the segmentation on the original image.

8.4.2 Weight Estimation

We compared our proposed approach with two existing methods for prawn weight estimation: the traditional linear regression method [199], and a Deep Learning-based method [437] based on shape or contour features for weight estimation. In the shape or contour features method for prawn weight estimation, the prawn body is segmented from the image using a threshold segmentation process. This process involves selecting a threshold value that separates the pixels in the image into two groups: foreground and background. The pixels with intensity values above the threshold are classified as foreground pixels, which belong to the prawn body, while the pixels with intensity values below the threshold are classified as background pixels. This is demonstrated in Fig. 8.3.

After segmenting the prawn body, the next step is to estimate its weight. One way to do this is to count the number of pixels in the segmented region, which is assumed to be proportional to the prawn weight. This pixel count is then correlated to the actual weight of the prawn using a linear regression model, e.g using a mathematical model similar to [199] for fish weight estimation. Another approach is to use Deep Learning, e.g. a neural network [437] that predicts the prawn's weight based on the pixel count. However, these methods have some limitations. For example, it assumes that the relationship between pixel count and prawn weight is linear. Additionally, this method does not take into account the shape and position of the prawn body, which can vary from one image to another.

These methods were compared against our proposed approach for prawn weight estimation, which involves generating 66 possible distances from the 12 landmarks and feeding them into a weight regression module. We generated a distance matrix as seen in Fig. 8.4 using the 12 predicted landmarks. This distance matrix contains the pairwise Euclidean distances between all possible pairs of the 12 landmarks, resulting in a total of $(12(12-1)/2 = 66)$ possible distances. These distances are then fed to the WRM, which uses a deep learning-based approach to predict the weight of the prawn. Our proposed approach of using the distance matrix as features for prawn weight estimation has several advantages over traditional methods such as the segmentation method. It takes into

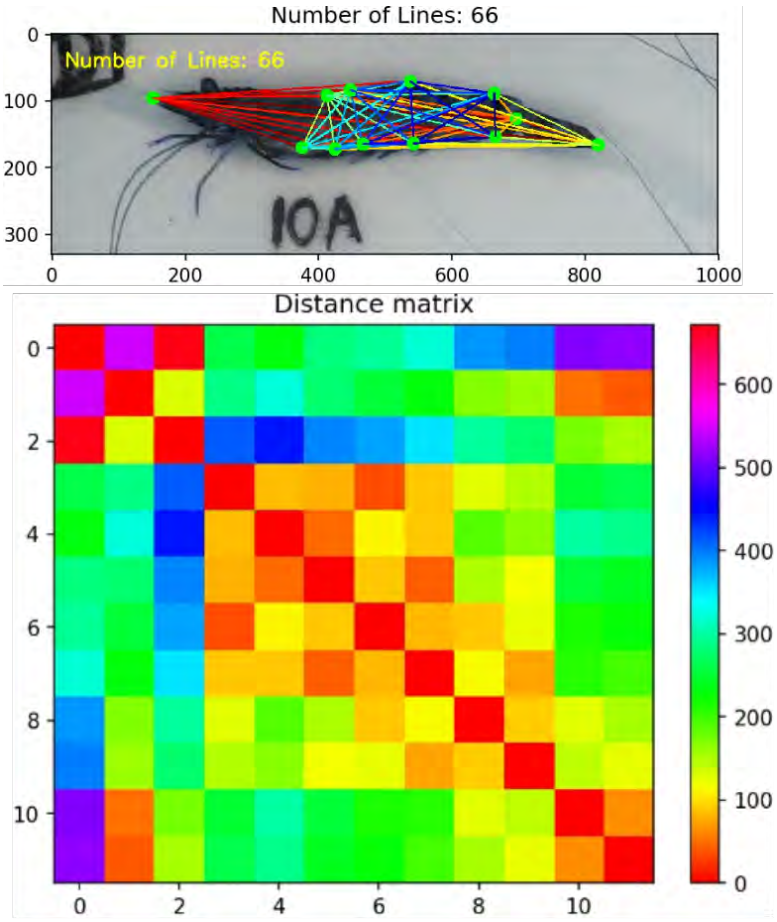

Figure 8.4: Top: Image of a prawn with the 66 possible distances between its 12 landmarks marked. Bottom: The resulting distance matrix plot computed from these distances.

account the spatial relationships between the landmarks, providing a more accurate estimation of prawn weight. Furthermore, our approach is less sensitive to variations in the prawn's posture or orientation, which can significantly affect the accuracy of traditional methods.

Table 8.5 shows the comparison results. Our proposed approach achieved the lowest mean absolute error (MAE) and mean squared error (MSE) values and the highest Coefficient of determination among the three methods. Specifically, our approach achieved an MAE of 0.649 g, an MSE of 0.986 g, and a Coefficient of determination of 0.934, which outperformed the other two methods. These results demonstrate the effectiveness and superiority of our proposed approach for prawn weight estimation.

In addition to evaluating the overall performance of our proposed weight estimation approach, we also used visualizations to further understand our model's performance and

Table 8.5: Comparison of prawn weight estimation methods using mean absolute error (MAE) in grams, mean squared error (MSE) in grams, and coefficient of determination (R^2)

Method	**MAE (g)**	**MSE (g)**	**Coefficient of determination**
Linear Regression	0.893	1.848	0.825
Deep Learning-based Method	0.880	1.713	0.896
Proposed Approach	**0.630**	**0.735**	**0.952**

identify areas for improvement. Specifically, we used Fig. 8.5 to visualize the relationships between the predicted and the true weight values. The plot in Fig. 8.5 shows the relationship between predicted weight and true weight for three different methods: Linear Regression, Deep Learning-based Method, and the Proposed Approach. The plot can provide information about the accuracy and precision of the different methods. For example, if the points on the plot fall close to the diagonal line ($y = x$), then the predicted weights are close to the true weights, indicating a high level of accuracy. Additionally, if the points are tightly clustered around the diagonal line, then the method is precise in its predictions.

Based on the plots in Fig. 8.5, it appears that the Proposed Approach has a higher correlation between predicted and true weight values compared to the Linear Regression and Deep Learning-based methods. This can be seen by the tighter clustering of points around the line of best fit. The presence of outliers in the Linear Regression and Deep Learning-based methods may be affecting the overall correlation.

Outliers can affect correlation in a number of ways. They can either increase or decrease the correlation coefficient depending on their location relative to the regression line. An outlier that is near where the regression line might normally go increases the correlation value, while an outlier away from the regression line decreases it.

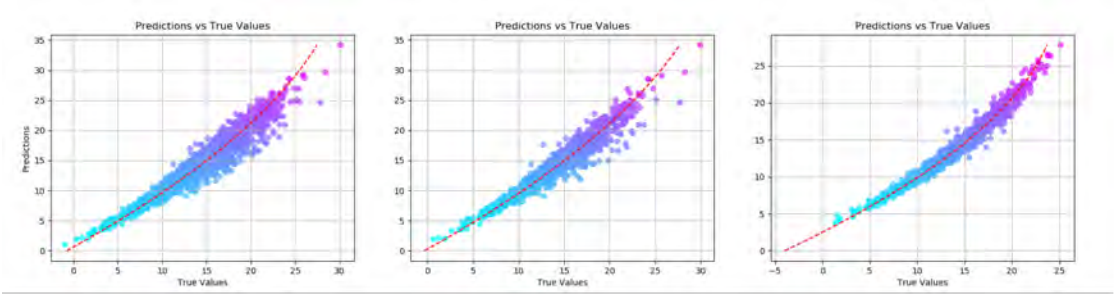

Figure 8.5: Plot showing the relationship between predicted weight and true weight for three different methods. From left to right: results for the Linear Regression method, Deep Learning-based Method, and our Proposed Approach.

8.4.3 Ablation Study on Weight Estimation

Our proposed model uses 66 distances as its features to predict the prawn weights. To investigate the impact of dimensionality reduction on our proposed approach, we conducted an ablation study using Principal Component Analysis (PCA). The purpose of PCA was to reduce the high number of dimensions/features per observation, which in our case were the 66 possible distances between the 12 landmarks marked on a prawn. We applied PCA to the distance matrix and reduced the number of components to 2, 5, and 10. We then used the reduced feature sets to predict the weight of prawns. The results are shown in Table 8.6.

Surprisingly, we found that using the full set of 66 distances without PCA resulted in better weight estimation results. This suggests that PCA was not necessary for this particular task and that the high-dimensional feature space was important for accurately capturing the information needed for prawn weight estimation. This finding is also in agreement with the observation shown in Table 8.6, which demonstrates the more features used, the lower the MAE and MSE of the model.

Table 8.6: Comparison of weight estimation results using PCA with a different number of components.

Method	**MAE (g)**	**MSE (g)**	**Coefficient of determination**
PCA ($n = 2$)	0.784	0.851	0.915
PCA ($n = 5$)	0.765	0.842	0.924
PCA ($n = 10$)	0.724	0.816	0.931
No PCA	**0.630**	**0.735**	**0.952**

8.4.4 Morphometric Analyses

In our proposed deep learning model, we detect keypoints (landmarks) on the prawn body to achieve highly accurate weight estimations. By detecting and analysing these landmarks, other important physical characteristics of the prawn, such as its length, width, and shape, can be extracted. The best part is that this morphometric analysis is essentially a byproduct of the weight estimation process. This represents a significant value-add for aquaculture and fisheries managers, who can now obtain valuable morphometric information about their prawn populations without incurring additional expenses or resources.

Morphometric analyses refer to the measurement of various physical features of an organism, such as length, width, and area. These measurements are essential for understanding the biology of the organism and can be used to identify and classify different

species while understanding their growth rate and possible body deformities. Morphometric information can also be used to study and identify specific traits that are desirable for selective breeding or commercial purposes.

Traditionally, morphometric analyses involve having a human operator measure and manually record various morphology traits on the animal body. This is a slow and inefficient process that cannot be easily scaled. Automated morphometric analyses facilitated using deep-learning-based computer vision and image processing, on the other hand, offer several benefits over traditional manual measurements. Firstly, they are much faster and more efficient, allowing researchers and operators to process larger stock/data quantities in a shorter amount of time. This is particularly important when dealing with large populations/datasets, as manual measurements are time-consuming and error-prone. Additionally, automated morphometric analyses are more objective, as they are not subject to human bias or error.

We performed a morphometric shape analysis on five important prawn traits derived from landmark data. These traits are shown in Table 8.2 and include total length, body length, the first abdominal segment height (First ASH), the third abdominal segment height(Third ASH), and the last abdominal segment height (Last ASH). We used the landmark data obtained from our model to calculate these five traits for each individual prawn and plotted their distributions in Fig. 8.6. We also computed the correlation matrix among the traits and visualized it as a heatmap in Fig. 8.7. The results show that total length and body length are highly correlated ($r = 0.99$), and so are the First and Third, and Third and Last ASH ($r = 0.93$). The other traits had high correlations as well. These patterns reflect the variation in shape and size among the prawns.

The accuracy of measurements is critical for the quantitative comparison of prawn morphometry, where accuracy refers to the closeness of measurements to the true value. To assess accuracy, this study utilized the mean absolute difference (MAD) between manual and DL measurements.

The MAD is a measure of accuracy that calculates the average absolute deviation between two values. It is obtained by dividing the sum of absolute deviations between each value by the number of values. The mathematical expression for MAD is:

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|(x_i - y_i)|$$

where n is the total number of observations, x_i and y_i are the values of the $i-th$ observation in two different samples, and the vertical bars denote absolute value.

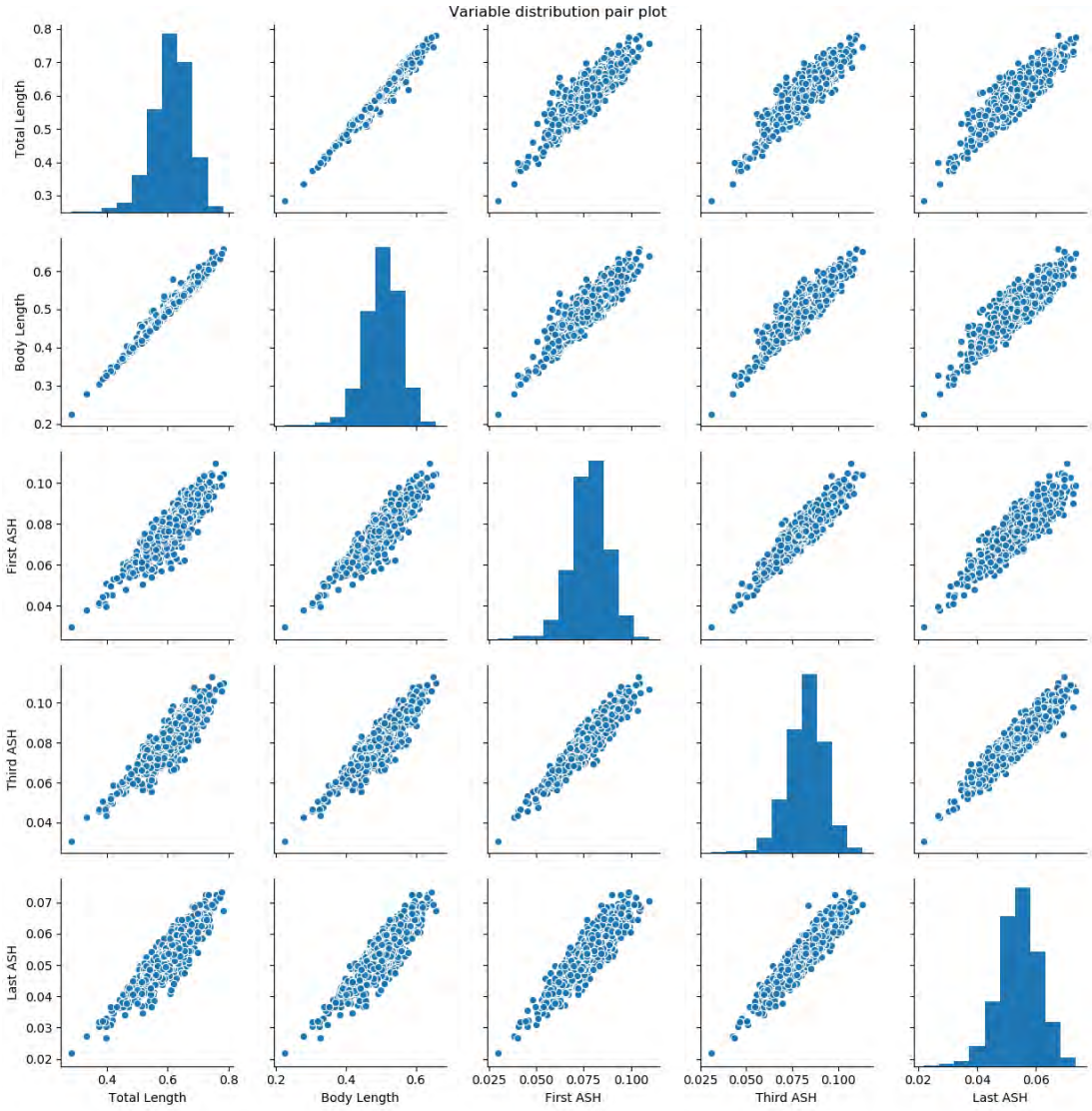

Figure 8.6: The distribution pair plots for the five important prawn traits from Table 8.2, i.e. Total length, Body length, First abdominal segment height ”First ASH”, Third abdominal segment height ”Third ASH”, Last abdominal segment height ”Last ASH”

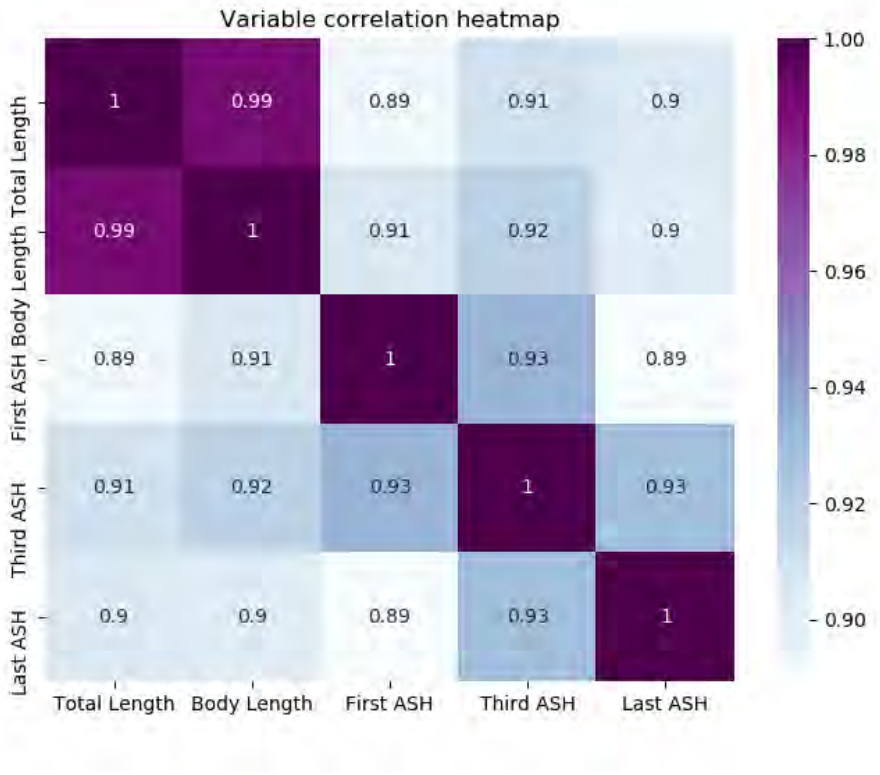

Figure 8.7: The correlation heatmap for the five important prawn traits from Table 8.2, i.e. Total length, Body length, First abdominal segment height ”First ASH”, Third abdominal segment height ”Third ASH”, Last abdominal segment height ”Last ASH”

Table 8.7 presents a performance comparison of various deep learning (DL) networks in terms of their ability to accurately measure morphometric features. The table shows the mean absolute difference (MAD) in millimetres between manual and DL measurements for total length, body length, first ASH, third ASH, and last ASH. The results indicate that our proposed KPFEM-based network outperforms other networks such as U-net, ResNet-18, ShuffleNet-v2, MobileNet-v2, and SqueezeNet in terms of MAD for all morphometric features. This suggests that our approach is more accurate in measuring morphometric features compared to other DL networks used for landmark detection.

Table 8.7: Morphometric Analyses Performance Comparison: Mean Absolute Difference (MAD) between Manual and DL Measurements (mm)

Network	Total length	Body length	First ASH	Third ASH	Last ASH
U-net [32]	1.81	1.83	1.73	1.72	1.51
ResNet-18 [226]	1.72	1.85	1.71	1.64	1.43
ShuffleNet-v2 [414]	2.34	2.22	2.15	2.15	2.02
MobileNet-v2 [415]	2.23	2.24	2.04	2.12	1.94
SqueezeNet [416]	2.14	1.93	1.94	1.85	1.74
KPFEM (ours)	**1.61**	**1.54**	**1.53**	**1.45**	**1.31**

8.4.5 Principal Component Analysis (PCA)

In addition to our aforementioned analysis, we also performed PCA on the predicted landmarks from our model to analyze the morphometric shape of prawns. PCA is a statistical technique commonly used for dimensionality reduction and identifying patterns in data. By reducing the dimensionality of the data, PCA allows us to identify the principal components (PCs) that explain the most variability in the data.

We chose to use PCA in our analysis as a data-driven approach to analyze prawn shape variation in the predicted landmark data. Even though we have ground truth measurements of body size for the prawns, PCA allows us to understand the PCs and explain variability in the data and how it represents the major sources of shape variation in prawns. This provides a complementary and holistic view of the shape variation beyond individual distance measurements.

It is important to note that the results obtained from PCA are based on the predicted landmark data, which may have some inaccuracies. However, we believe that the predicted landmark data still provide valuable insights into the shape variation among prawns

and can be used as a proxy for understanding the morphometric characteristics of the prawns. This is especially important because collecting manual landmarks is impractical at large scales.

The PCA analysis revealed that the first two principal components (PC1, PC2) accounted for 94.2% of the total variability in the data, while the next four PCs, accounted for only 2.6% of variability. Detailed variability results are shown in Table 8.8.

Table 8.8: Explanation of Variability by Various Principal Components (PCs)

	PC1	PC2	PC3	PC4	PC5	PC6
Standard deviation	191.24	23.63	19.22	14.78	11.42	10.17
Proportion of Variance	0.916	0.026	0.014	0.006	0.004	0.002
Cumulative Proportion	0.916	0.942	0.956	0.962	0.966	0.968

The results for the first two PCs are also shown in Fig. 8.8. In the left panel, the circles represent individual prawns with their dataset ID shown next to them. Their positions on the plot are determined by their scores on PC1 (Dim1) and PC2 (Dim2). The cos2 values measure the quality of representation of the individuals by the principal components.

PC1 accounted for the highest proportion of the variation (91.6%) among the landmarks and was used as a representation of the overall size of the prawns. The positive loadings of PC1 indicate that an increase in PC1 scores corresponds to an increase in the distance between landmarks 1 and 3, which represents the total length of the prawn. PC1 can be used as an indicator of overall size, and the higher variance accounted for by PC1 may be due to the presence of a wide range of sizes in the samples.

PC2 (Dim2) explained 2.6% of the total variation among landmarks and represents the proportionate body width/shape of the prawns. The extremes on the PC2 axis on Fig. 8.8(left) represent a very thin/elongated body shape at the low end and a thick/fatty shape at the high end.

Figure 8.8)(right) illustrates how different variables (distances) contribute to the variation in shape among prawns with respect to the two main PCs, i.e. PC1 and PC2. Here, each of the arrows shows one of the distances, e.g. d_1_5 designates the distance between landmark 1 and 5, and how it relates to the two PCs.

In conclusion, our PCA analysis of the predicted landmark data revealed that PC1 represents the overall size and PC2 represents the proportionate body width/shape of the prawns. These findings provide important insights into our targeted morphometric characteristics of prawns and lay a foundation for further research on the genetic and environmental factors that affect prawn morphology [438].

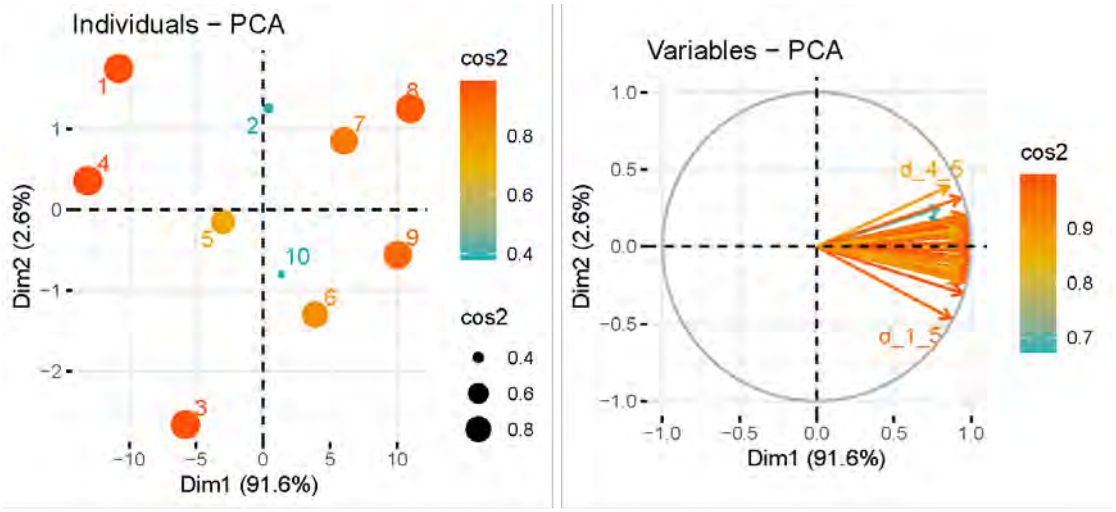

Figure 8.8: PCA Analysis. In the left panel, the circles represent individual prawns with their dataset ID shown next to them. Their positions on the plot are determined by their scores on PC1 (Dim1) and PC2 (Dim2). The cos2 values measure the quality of representation of the individuals by the principal components. The right panel illustrates how different variables (distances) contribute to the variation in shape among prawns with respect to the two main PCs. Here, each of the arrows shows one of the distances, e.g. d_1_5 designates the distance between landmark 1 and 5, and how it relates to the two PCs.

8.5 Discussion

Fish and prawns have distinct morphological characteristics that can affect the accuracy of landmark detection. For example, fish have a more elongated body shape and a variety of fin shapes and sizes, while prawns have a more compact body shape and distinctive features such as antennae and claws. These differences in morphology can make it challenging to accurately detect landmarks using a single model that is designed to work for both fish and prawns.

To address this challenge, separate models were developed for fish and prawn landmark detection tasks. These models were specifically designed to account for the unique morphological characteristics of each species, allowing for more accurate landmark detection. By developing separate models, we were able to improve the performance of our landmark detection system and provide more reliable results for both fish and prawn species. The accurate estimation of weight and other morphological features of individuals being farmed in aquaculture is crucial data that when acquired on industrial scales can improve crop manage- ment, support decision-making, and be embedded in product grading and processing activities. Traditional methods of monitoring and acquiring morphological

data from aquaculture animals are laborious, hence the need for developing automated methods using computer vision and deep learning. In this paper, we proposed a novel approach that uses a Kronecker product-based feature extraction module and a landmark localization module. We utilized these efficient modules to extract important landmarks on the prawn body in a highly accurate, fast, and efficient way to perform accurate weight estimation and morphometric analysis of prawns from images. Our work is the first that applies the Kronecker product operation for feature extraction in morphometrics analysis and uses a localization network for landmark detection in prawn images.

Our deep learning-based image processing approach can be a valuable tool for improving the efficiency and profitability of aquaculture and fisheries operations, while also promoting sustainable practices and minimizing environmental impacts. Our approach helps researchers and managers obtain accurate measurements of prawn size, weight, and other morphological features without invasive or time-consuming manual methods. This allows potentially for more efficient monitoring of populations, aiding in informed decisions about stocking densities, feeding regimes, and harvesting schedules. Noninvasive and accurate weight data can also provide critical information for managing prawn populations and predicting future yields, enhancing decision support.

Our approach has several strengths. First, it is fully automated, which saves time and reduces manual labour. Secondly, it is accurate and robust, enabling reliable prawn weight estimation even in challenging environments. Thirdly, our use of Deep Learning techniques to extract features from prawn images is a promising area of research that has shown great potential for morphometric analyses.

However, our approach also has limitations. It heavily relies on the quality of input images, and low-quality images may not provide enough information for accurate prawn weight estimation. Additionally, large-scale annotated datasets are needed for training our Deep Learning architecture, which may be difficult to obtain in some situations.

In future work, we aim to address these limitations by exploring new approaches to handle low-quality images and developing efficient and automated methods for acquiring and annotating large-scale datasets. Furthermore, we will investigate the application of our approach to other species of aquatic animals and explore the potential for using it for other applications such as automated disease diagnosis and monitoring. We believe that our work can significantly contribute to prawn aquaculture management and production, opening up new opportunities for Deep Learning applications in aquatic animal research and aquaculture engineering.

8.6 Conclusions

We proposed a novel Deep Learning approach for automated morphometric analyses and weight estimation of prawns from images. Our approach consisted of two main modules: a Kronecker product-based feature extraction module that efficiently combined low-level and high-level features to be used in the second landmark localization module to accurately detect landmarks on the prawn body. We showed that the use of Kronecker product-based feature extraction can lead to significant model improvements compared to conventional convolution-based feature extractors. We also demonstrated that this resulted in a significantly higher image processing throughput for our model, compared to the state-of-the-art convolution-based models.

We evaluated our approach on a large-scale dataset of farmed prawn images. We applied our approach to two important applications, i.e. weight estimation, and morphometric analyses of prawns. Our experimental results showed that our approach outperformed existing methods in terms of accuracy, robustness, and efficiency. The proposed approach provided accurate weight estimation and morphometric analysis, which are critical for prawn aquaculture management and production.

8.6.1 CO2 Emission Related to Experiments

Experiments were conducted using a private infrastructure, which has a carbon efficiency of 0.432 kgCO_2eq/kWh. A cumulative of 500 hours of computation was performed on the hardware of type RTX 2080 Ti (TDP of 250W). Total emissions are estimated to be 54 kgCO_2eq of which 0 percents were directly offset. Estimations were conducted using the MachineLearning Impact calculator presented in [439].

More in detail, in Table 8.4, we compare our proposed model with a ResNet-18 for landmark detection. We find that our model reduces both training time and carbon emissions by 25%. The ResNet-18 takes about 20 hours and emits 2.16 kgCO_2eq, while our model takes about 15 hours and emits 1.62 kgCO_2eq. Carbon emissions are a major concern for training large deep-learning models. Therefore, we believe that our method is a small step towards more efficient and eco-friendly models.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

This thesis presented several novel DL methods for underwater image processing and analysis. The main contributions of this thesis are:

- A comprehensive review of existing Computer Vision and DL research on fish identification in marine environments from 2003 to 2021.
- Two novel methods for fish segmentation and tracking in underwater videos using self-supervised learning and optical flow models.
- A novel framework called UDnet that adapts to Uncertainty Distribution in its unsupervised reference map generation to produce enhanced underwater images.
- Two novel DL architectures for fish landmark detection and prawn weight estimation from images using convolution operations based on Vision Transformers and Kronecker product operation.

The experimental results showed that the proposed methods achieved state-of-the-art performance on various underwater image datasets and demonstrated their effectiveness, robustness, efficiency, and generalisability. The methods also addressed some of the challenges of applying DL to underwater image processing such as data scarcity, annotation cost, domain adaptation, and uncertainty handling. The methods also provided some insights into the field of marine habitat monitoring and suggested some future directions for using DL for underwater image processing. This thesis aimed to bridge the gap between DL and underwater image processing, and to facilitate the advancement of both fields. The main contributions and findings of this thesis are summarized as follows:

In Chapter 2, a review of existing Computer Vision and Deep Learning research on fish identification in marine environments from 2003 to 2021 was provided. Key concepts of Deep Learning were overviewed and various studies that used this technique were evaluated. The main challenges of applying Deep Learning to underwater image processing were discussed and some solutions were suggested. Some insights into the field of marine habitat monitoring were offered and some future directions for using Deep Learning for underwater image processing were proposed.

In Chapter 3, an exploration of how DL techniques can be applied to monitor underwater fish habitats was provided. A tutorial on the key concepts and steps of DL development for marine scientists who want to learn and apply DL to their own problems was provided. A comprehensive survey of existing DL methods for fish habitat monitoring tasks such as classification, counting, localisation, and segmentation was conducted. The performance and limitations of various DL techniques using publicly available underwater fish datasets were compared. Some open challenges and opportunities for future research in this domain were discussed.

In Chapters 4 and 5, *the first research question was addressed*. Two novel methods for fish segmentation and tracking in underwater videos were presented. The first method used a Transformer-based model that learned from self-supervision on unlabelled videos. The model achieved high-quality fish segmentation in underwater videos captured in situ in the wild. The model outperformed previous self-supervised methods and was close to supervised methods on two new unseen underwater video datasets. The model also showed great compute-efficiency and generalisability. The second method proposed a three-stage framework that used an optical flow model to generate pseudo labels based on spatial and temporal consistency between frames. A self-supervised model refined the pseudo-labels incrementally. The refined labels were used to train a segmentation network. No human annotations were required during the training or inference. The method was validated on three public underwater video datasets and demonstrated its effectiveness and robustness.

In Chapter 6, *the second research question was addressed*. A novel framework called Uncertainty Distribution Network (UDnet) that adapts to Uncertainty Distribution in its unsupervised reference map generation to produce enhanced output images was introduced. This chapter addressed the second research question. UDnet consists of three main parts: a statistically guided multi-colour space stretch module, a U-Net-like conditional variational autoencoder module, and a probabilistic adaptive instance normalization block. UDnet does not require manual human annotation and can learn with a limited

amount of data to achieve state-of-the-art results. UDnet was evaluated on eight publicly-available datasets and showed competitive performance compared to other state-of-the-art approaches.

Finally, in Chapters 7 and 8, *the third and last research question was addressed*. Two novel DL architectures for fish landmark detection and prawn weight estimation from images were proposed. The first architecture was called Mobile Fish Landmark Detection network (MFLD-net), which used convolution operations based on Vision Transformers. MFLD-net could achieve keypoint detection accuracies on par or better than some of the state-of-the-art CNNs on a fish image dataset. MFLD-net was lightweight and suitable for embedded and mobile devices, and could learn with a limited amount of data. MFLD-net also showed great generalisation capabilities. The second architecture was a novel DL approach for automated weight estimation and morphometric analysis of prawns from images. The approach consisted of three main components: a feature extraction module that leveraged the Kronecker product operation to combine low-level and high-level features, a landmark localization module that predicted the coordinates of key points on the prawn body using a localization network, and a weight regression module that estimated the prawn weight based on the extracted features and landmarks using a fully connected network. The approach was evaluated on a large-scale dataset of prawn images collected from various farms and environments. The experimental results demonstrated that the approach outperformed existing methods in terms of accuracy, robustness, and efficiency.

In conclusion, the proposed deep learning architectures provide new and effective tools for analyzing visual data in marine and aquaculture environments. I hope that the methods proposed in this thesis will inspire further research and development in this field, leading to new applications and discoveries.

9.2 Limitations and Research Scope

Limitations and assumptions are inherent in any research and can affect the generalizability or applicability of the research findings. In the context of this thesis, some potential limitations and assumptions that may affect the generalizability or applicability of the research findings could include:

- The datasets used to evaluate the proposed methods may not be representative of all underwater environments and scenarios, limiting the generalizability of the results to other datasets or scenarios.

- The proposed methods may rely on certain assumptions about the characteristics of the underwater environment, such as lighting conditions, water clarity, or fish behaviour, which may not hold in all scenarios.
- The performance of the proposed methods may be affected by factors such as the quality and quantity of training data, the choice of hyperparameters, or the architecture of the deep learning models, which may limit their applicability to other datasets or scenarios.

The chosen scope of the research is justified by the need to address specific challenges in marine and aquaculture computer vision, such as fish segmentation, trajectory tracking, image enhancement, landmark detection, weight estimation and morphometric analyses. These challenges are important for various applications such as fish monitoring, stock assessment, aquaculture management and environmental protection. By focusing on these specific challenges and proposing novel deep learning architectures to address them, this thesis aims to advance the field of marine and aquaculture computer vision and provide practical solutions that can facilitate various applications.

9.3 Future Work

The methods presented in this thesis have demonstrated the potential of deep learning architectures for various challenges and tasks related to marine and aquaculture computer vision. However, there are still many opportunities for further research and improvement in this field. Some of the possible directions for future work are:

- Extending the self-supervised methods for fish segmentation and tracking to other underwater objects such as coral reefs, algae, and marine debris. This would enable a more comprehensive analysis of underwater scenes and ecosystems, as well as facilitate tasks such as habitat monitoring, biodiversity assessment, and environmental impact evaluation.
- Exploring different ways of incorporating Uncertainty Distribution into the reference map generation process to improve the quality and diversity of the enhanced output images. This could involve using different types of uncertainty measures, such as aleatoric or epistemic uncertainty, or applying different strategies for sampling or combining uncertain reference maps. This would allow for more realistic and robust image enhancement under various underwater conditions.

- Applying the Vision Transformer-based architecture for fish landmark detection to other tasks such as fish species recognition, pose estimation, and behaviour analysis. This would leverage the advantages of Vision Transformers over convolutional neural networks, such as better scalability, generalization, and interpretability, for more complex and high-level underwater image processing and analysis tasks.
- Developing more efficient and accurate methods for prawn weight estimation and morphometric analysis using deep learning that can handle occlusion, deformation, and illumination variation. This could involve using different types of input data, such as depth maps or thermal images, or applying different techniques for data augmentation or regularization. This would improve the performance and reliability of prawn weight estimation and morphometric analysis systems for aquaculture management and quality control purposes.
- Evaluating the proposed methods on larger and more diverse underwater image datasets collected from different regions, seasons, and depths. This would test the scalability and robustness of the proposed methods across different underwater environments and scenarios, as well as provide more insights into their strengths and limitations.

In addition to these specific directions based on the methods presented in this thesis, there are also some general directions that could be explored in future research on underwater image processing and analysis using deep learning:

- Investigating the potential of using multi-modal data such as acoustic signals, depth maps, and temperature sensors to complement the visual information for underwater image processing and analysis.
- Incorporating additional modalities: The use of multiple modalities, such as sound and environmental data, can provide complementary information that enhances the accuracy and robustness of the proposed methods.
- Addressing occlusions and complex background conditions: The proposed methods can be extended to address occlusions and complex background conditions, which are common challenges in underwater environments.
- Considering multi-species interactions: The proposed methods can be applied to analyze interactions between multiple species, which can provide valuable insights into ecological dynamics and environmental changes.

- Expanding the scope of applications: The proposed methods can be applied to a wide range of marine and aquaculture applications, such as underwater robotics, ocean exploration, and marine conservation.
- Improving interpretability and explainability: The proposed methods can be further improved to enhance interpretability and explainability, which are crucial for gaining insights and understanding the underlying mechanisms of the analyzed phenomena.

Overall, I believe that the proposed deep learning architectures have great potential for advancing the field of marine and aquaculture computer vision, and I look forward to seeing further research and development in this area.

Bibliography

[1] A. Saleh, M. Sheaves, and M. Rahimi Azghadi, "Computer vision and deep learning for fish classification in underwater habitats: A survey," *Fish and Fisheries*, vol. 23, no. 4, pp. 977–999, 7 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/faf.12666

[2] A. Saleh, M. Sheaves, D. Jerry, and M. R. Azghadi, "Applications of Deep Learning in Fish Habitat Monitoring: A Tutorial and Survey," *Arxiv*, 6 2022. [Online]. Available: http://arxiv.org/abs/2206.05394

[3] ——, "Transformer-based Self-Supervised Fish Segmentation in Underwater Videos," *Arxiv*, 6 2022. [Online]. Available: http://arxiv.org/abs/2206.05390

[4] ——, "Unsupervised Fish Trajectory Tracking and Segmentation," *Arxiv*, 8 2022. [Online]. Available: https://arxiv.org/abs/2208.10662v1

[5] ——, "Adaptive Uncertainty Distribution in Deep Learning for Unsupervised Underwater Image Enhancement," *Arxiv*, 12 2022. [Online]. Available: https://arxiv.org/abs/2212.08983v1

[6] A. Saleh, D. Jones, D. Jerry, and M. R. Azghadi, "A lightweight Transformer-based model for fish landmark detection," *Arxiv*, 9 2022. [Online]. Available: https://arxiv.org/abs/2209.05777v1

[7] A. Saleh, M. M. Hasan, H. W. Raadsma, M. S. Khatkar, D. Jerry, and M. R. Azghadi, "Prawn Morphometrics and Weight Estimation from Images using Deep Learning for Landmark Localization," *Arxiv*, 9 2022. [Online]. Available: https://arxiv.org/abs/2209.05777v1

[8] A. Saleh, I. H. Laradji, C. Lammie, D. Vazquez, C. A. Flavell, and M. R. Azghadi, "A Deep Learning Localization Method for Measuring Abdominal Muscle Dimensions in Ultrasound Images," *IEEE Journal of Biomedical and*

Health Informatics, vol. 25, no. 10, pp. 3865–3873, 10 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9444630/

[9] I. H. Laradji, A. Saleh, P. Rodriguez, D. Nowrouzezahrai, M. R. Azghadi, and D. Vazquez, "Weakly supervised underwater fish segmentation using affinity LCFCN." *Scientific reports*, vol. 11, no. 1, p. 17379, 12 2021. [Online]. Available: https://www.nature.com/articles/s41598-021-96610-2http://www.ncbi.nlm.nih.gov/pubmed/34462458http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC8405733

[10] E. M. Ditria, R. M. Connolly, E. L. Jinks, and S. Lopez-Marcano, "Annotated Video Footage for Automated Identification and Counting of Fish in Unconstrained Seagrass Habitats," *Frontiers in Marine Science*, vol. 8, 3 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fmars.2021.629485/full

[11] A. A. Jabri, A. Owens, and A. A. Efros, "Space-time correspondence as a contrastive random walk," in *Advances in Neural Information Processing Systems*, 2020.

[12] N. Araslanov, S. Schaub-Meyer, and S. Roth, "Dense Unsupervised Learning for Video Segmentation," *IEEE*, 11 2021. [Online]. Available: https://arxiv.org/abs/2111.06265v1

[13] Z. Lai, E. Lu, and W. Xie, "MAST: A memory-augmented self-supervised tracker," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.

[14] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy, "Tracking Emerges by Colorizing Videos," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11217 LNCS, pp. 402–419. [Online]. Available: https://link.springer.com/10.1007/978-3-030-01261-8_24

[15] Z. Lai and W. Xie, "Self-supervised Learning for Video Correspondence Flow," *30th British Machine Vision Conference 2019, BMVC 2019*, 5 2019. [Online]. Available: http://arxiv.org/abs/1905.00875

[16] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, and T. Huang, "YouTube-VOS: Sequence-to-Sequence Video Object Segmentation," in *Lecture*

Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018.

[17] 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, 12 2020. [Online]. Available: https://www.nature.com/articles/s41598-020-71639-x

[18] M. J. Islam, Y. Xia, and J. Sattar, "Fast Underwater Image Enhancement for Improved Visual Perception," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3227–3234, 4 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9001231/

[19] M. Jahidul Islam, P. Luo, and J. Sattar, "Simultaneous Enhancement and Super-Resolution of Underwater Imagery for Improved Visual Perception," in *Robotics: Science and Systems XVI*. Corvalis, Oregon, USA: Robotics: Science and Systems Foundation, 7 2020. [Online]. Available: http://www.roboticsproceedings.org/rss16/p018.pdf

[20] C. Li, C. Guo, W. Ren, R. Cong, J. Hou, S. Kwong, and D. Tao, "An Underwater Image Enhancement Benchmark Dataset and Beyond," *IEEE Transactions on Image Processing*, vol. 29, pp. 4376–4389, 2020.

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[22] D. M. Chandler, "Most apparent distortion: full-reference image quality assessment and the role of strategy," *Journal of Electronic Imaging*, vol. 19, no. 1, p. 011006, 1 2010. [Online]. Available: http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.3267105

[23] W. Xue, L. Zhang, X. Mou, and A. C. Bovik, "Gradient Magnitude Similarity Deviation: A Highly Efficient Perceptual Image Quality Index," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 684–695, 2 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6678238/

[24] 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, 12 2020. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/32887922http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC7473859https://www.nature.com/articles/s41598-020-71639-x

[25] T. Mandel, M. Jimenez, E. Risley, T. Nammoto, R. Williams, M. Panoff, M. Ballesteros, and B. Suarez, "Detection confidence driven multi-object tracking to recover reliable tracks from unreliable detections," *Pattern Recognition*, vol. 135, p. 109107, 3 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0031320322005878

[26] S. LopezMarcano, E. Jinks, C. A. Buelow, C. J. Brown, D. Wang, B. Kusy, E. Ditria, and R. M. Connolly, "Automatic detection of fish and tracking of movement for ecology," *Ecology and Evolution*, vol. 11, no. 12, pp. 8254–8263, 6 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/ece3.7656

[27] R. Liu, X. Fan, M. Zhu, M. Hou, and Z. Luo, "Real-world underwater enhancement: Challenges, benchmarks, and solutions under natural light," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 12, pp. 4861–4875, 2020.

[28] 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," *http://arxiv.org/abs/2004.01241*, 2020. [Online]. Available: http://arxiv.org/abs/2004.01241

[29] K. Panetta, C. Gao, and S. Agaian, "Human-visual-system-inspired underwater image quality measures," *IEEE J. Ocean. Eng.*, vol. 41, no. 3, pp. 541–551, 7 2016.

[30] J. Ke, Q. Wang, Y. Wang, P. Milanfar, and F. Yang, "MUSIQ: Multi-scale Image Quality Transformer," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5128–5137, 8 2021. [Online]. Available: https://arxiv.org/abs/2108.05997v1

[31] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a Completely Blind Image Quality Analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 3 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6353522/

[32] 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*, 2015, pp. 234–241.

[33] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-Scale Conv-Attentional Image Transformers," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 10 2021, pp. 9961–9970. [Online]. Available: https://ieeexplore.ieee.org/document/9710209/

[34] Georgiou Giannis, "Mediterranean Fish Species," 2021. [Online]. Available: https://www.kaggle.com/datasets/giannisgeorgiou/fish-species

[35] D. R. Jerry and S. C. Cairns, "Morphological variation in the catadromous Australian bass, from seven geographically distinct riverine drainages," *Journal of Fish Biology*, vol. 52, no. 4, pp. 829–843, 1998.

[36] X. Yang, S. Zhang, J. Liu, Q. Gao, S. Dong, and C. Zhou, "Deep learning for smart fish farming: applications, opportunities and challenges," *Reviews in Aquaculture*, vol. 13, no. 1, pp. 66–90, 1 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/raq.12464

[37] S. Zarco-Perello and S. Enríquez, "Remote underwater video reveals higher fish diversity and abundance in seagrass meadows, and habitat differences in trophic interactions." *Scientific reports*, vol. 9, no. 1, p. 6596, 12 2019. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/31036932http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6488625

[38] C. V. Piggott, M. Depczynski, M. Gagliano, and T. J. Langlois, "Remote video methods for studying juvenile fish populations in challenging environments," *Journal of Experimental Marine Biology and Ecology*, vol. 532, p. 151454, 11 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0022098119304319

[39] K. L. Pope, S. E. Lochmann, and M. K. Young, "Methods for Assessing Fish Populations," *Inland fisheries management in North America*, 2010.

[40] R. S. Rasmussen and M. T. Morrissey, "Methods for the Commercial Fish and Seafood Species," *Comprehensive reviews in food science and food safety*, 2008.

[41] E. B. Thorstad, A. H. Rikardsen, A. Alp, and F. Okland, "The Use of Electronic Tags in Fish Research - An Overview of Fish Telemetry Methods," *Turkish Journal of Fisheries and Aquatic Sciences*, 2013.

[42] D. A. Konovalov, A. Saleh, M. Bradley, M. Sankupellay, S. Marini, and M. Sheaves, "Underwater Fish Detection with Weak Multi-Domain Supervision," in *2019 International Joint Conference on Neural Networks (IJCNN)*, vol. 2019-July. IEEE, 7 2019, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8851907/

[43] R. Hilborn and C. J. Walters, "Quantitative fisheries stock assessment: Choice, dynamics and uncertainty," *Reviews in Fish Biology and Fisheries*, vol. 2, no. 2, pp. 177–178, 6 1992. [Online]. Available: http://link.springer.com/10.1007/BF00042883

[44] B. Zion, "The use of computer vision technologies in aquaculture A review," *Computers and Electronics in Agriculture*, vol. 88, pp. 125–132, 10 2012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0168169912001950

[45] A. Olsen, D. A. Konovalov, B. Philippa, P. Ridd, J. C. Wood, J. Johns, W. Banks, B. Girgenti, O. Kenny, J. Whinney, B. Calvert, M. R. Azghadi, and R. D. White, "DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning," *Scientific Reports*, vol. 9, no. 1, p. 2058, 12 2019. [Online]. Available: https://www.nature.com/articles/s41598-018-38343-3http://www.nature.com/articles/s41598-018-38343-3

[46] A. Saleh, I. H. Laradji, C. Lammie, D. Vazquez, C. A. Flavell, and M. R. Azghadi, "A Deep Learning Localization Method for Measuring Abdominal Muscle Dimensions in Ultrasound Images," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 10, pp. 3865–3873, 10 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9444630/

[47] M. R. Azghadi, C. Lammie, J. K. Eshraghian, M. Payvand, E. Donati, B. Linares-Barranco, and G. Indiveri, "Hardware Implementation of Deep Network Accelerators towards Healthcare and Biomedical Applications," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 6, pp. 1138–1159, 12 2020.

[48] H. Zheng, R. Wang, W. Ji, M. Zong, W. K. Wong, Z. Lai, and H. Lv, "Discriminative deep multi-task learning for facial expression recognition,"

Information Sciences, vol. 533, pp. 60–71, 9 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0020025520303601

[49] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, and B. Hodjat, "Evolving Deep Neural Networks," in *Artificial Intelligence in the Age of Neural Networks and Brain Computing*. Elsevier, 1 2019, pp. 293–312. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128154809000153

[50] G. Montavon, W. Samek, and K. R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2 2018.

[51] M. Willi, R. T. Pitman, A. W. Cardoso, C. Locke, A. Swanson, A. Boyer, M. Veldthuis, and L. Fortson, "Identifying animal species in camera trap images using deep learning and citizen science," *Methods in Ecology and Evolution*, vol. 10, no. 1, pp. 80–91, 1 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/2041-210X.13099

[52] M. A. Tabak, M. S. Norouzzadeh, D. W. Wolfson, S. J. Sweeney, K. C. Vercauteren, N. P. Snow, J. M. Halseth, P. A. Di Salvo, J. S. Lewis, M. D. White, B. Teton, J. C. Beasley, P. E. Schlichting, R. K. Boughton, B. Wight, E. S. Newkirk, J. S. Ivan, E. A. Odell, R. K. Brook, P. M. Lukacs, A. K. Moeller, E. G. Mandeville, J. Clune, and R. S. Miller, "Machine learning to classify animal species in camera trap images: Applications in ecology," *Methods in Ecology and Evolution*, 2019.

[53] A. Saleh, M. Sheaves, and M. R. Azghadi, "Computer Vision and Deep Learning for Fish Classification in Underwater Habitats: A Survey," *Wiley Online*, 3 2022. [Online]. Available: http://arxiv.org/abs/2203.06951http://dx.doi.org/10.1111/faf.12666

[54] L. Xu, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "Deep learning for marine species recognition," in *Smart Innovation, Systems and Technologies*. Springer Science and Business Media Deutschland GmbH, 2019, vol. 136, pp. 129–145. [Online]. Available: http://link.springer.com/10.1007/978-3-030-11479-4_7

[55] L. Deng and D. Yu, "Deep learning: Methods and applications," 2013.

[56] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of Deep Learning for Object Detection," *Procedia Computer Science*, vol. 132, pp. 1706–1717, 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877050918308767

[57] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," 2017.

[58] M. Goodwin, K. T. Halvorsen, L. Jiao, K. M. Knausgård, A. H. Martin, M. Moyano, R. A. Oomen, J. H. Rasmussen, T. K. Sørdalen, and S. H. Thorbjørnsen, "Unlocking the potential of deep learning for marine ecology: overview, applications, and outlook," *ICES Journal of Marine Science*, vol. 79, no. 2, pp. 319–336, 9 2022. [Online]. Available: https://arxiv.org/abs/2109.14737v1

[59] D. Li and L. Du, "Recent advances of deep learning algorithms for aquacultural machine vision systems with emphasis on fish," *Artificial Intelligence Review*, pp. 1–40, 11 2021. [Online]. Available: https://link.springer.com/article/10.1007/s10462-021-10102-3https://link.springer.com/10.1007/s10462-021-10102-3

[60] A. H. J. Oomes, "Perception: Theory, Development and Organisation." *Optometry and Vision Science*, vol. 78, no. 7, p. 477, 7 2001. [Online]. Available: http://journals.lww.com/00006324-200107000-00005

[61] B. Wang and J. D. Weiland, "Visual system," in *Neuroprosthetics: Theory and Practice: Second Edition*, 2017.

[62] D. H. Ballard and C. M. Brown, *Computer Vision*. Prentice Hall, 1982. [Online]. Available: https://archive.org/details/computervision0000ball

[63] T. Huang, *Computer Vision : Evolution And Promise*. Geneva: CERN, 11 1996. [Online]. Available: http://cds.cern.ch/record/400313/files/p21.pdf

[64] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*. Thomson, 2008.

[65] A. O. Mader, C. Lorenz, M. Bergtholdt, J. von Berg, H. Schramm, J. Modersitzki, and C. Meyer, "Detection and localization of spatially correlated point landmarks in medical images using an automatically learned conditional random field," *Computer Vision and Image Understanding*, 2018.

[66] T. Hohage, B. Sprung, and F. Weidling, "Inverse Problems," in *Topics in Applied Physics*, 2020, pp. 145–164. [Online]. Available: http://link.springer.com/10.1007/978-3-030-34413-9_5

[67] M. Sarigül and M. Avci, "Comparison of Different Deep Structures for Fish Classification," *International Journal of Computer Theory and Engineering*, 2017.

[68] A. Salman, A. Jalal, F. Shafait, A. Mian, M. Shortis, J. Seager, and E. Harvey, "Fish species classification in unconstrained underwater environments based on deep learning," *Limnology and Oceanography: Methods*, vol. 14, pp. 570–585, 2016.

[69] H. Qin, X. Li, J. Liang, Y. Peng, and C. Zhang, "DeepFish: Accurate underwater live fish recognition with a deep architecture," *Neurocomputing*, vol. 187, pp. 49–58, 4 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231215017312

[70] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, 2017, pp. 843–852.

[71] A. Permaloff and C. Grafton, "Optical Character Recognition," *PS: Political Science and Politics*, vol. 25, no. 3, p. 523, 9 1992. [Online]. Available: https://www.jstor.org/stable/419444?origin=crossref

[72] J. K. Park, B. K. Kwon, J. H. Park, and D. J. Kang, "Machine learning-based imaging system for surface defect inspection," *International Journal of Precision Engineering and Manufacturing - Green Technology*, 2016.

[73] H. Trinh, Q. Fan, P. Gabbur, and S. Pankanti, "Hand tracking by binary quadratic programming and its application to retail activity recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.

[74] B. J. Erickson, P. Korfiatis, Z. Akkus, and T. L. Kline, "Machine Learning for Medical Imaging," *RadioGraphics*, vol. 37, no. 2, pp. 505–515, 3 2017. [Online]. Available: http://pubs.rsna.org/doi/10.1148/rg.2017160130

[75] F. Falcini, G. Lami, and A. M. Costanza, "Deep Learning in Automotive Software," *IEEE Software*, vol. 34, no. 3, pp. 56–63, 5 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7927925/

[76] A. Brunetti, D. Buongiorno, G. F. Trotta, and V. Bevilacqua, "Computer vision and deep learning techniques for pedestrian detection and tracking: A survey," *Neurocomputing*, 2018.

[77] S. Kim, B. Park, B. S. Song, and S. Yang, "Deep belief network based statistical feature learning for fingerprint liveness detection," *Pattern Recognition Letters*, vol. 77, pp. 58–65, 7 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167865516300198

[78] M. C. Chuang, J. N. Hwang, F. F. Kuo, M. K. Shan, and K. Williams, "Recognizing live fish species by hierarchical partial classification based on the exponential benefit," in *Proc. Int. Conf. Image Process.* Paris, France: IEEE, 10 2014, pp. 5232–5236.

[79] M. C. Chuang, J. N. Hwang, and K. Williams, "A feature learning and object recognition framework for underwater fish images," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1862–1872, 4 2016.

[80] D. S. Y. Kartika and D. Herumurti, "Koi fish classification based on HSV color space," in *Proceedings of 2016 International Conference on Information and Communication Technology and Systems, ICTS 2016*, 2017, pp. 96–100.

[81] B. Zion, V. Alchanatis, V. Ostrovsky, A. Barki, and I. Karplus, "Classification of guppies' (Poecilia reticulata) gender by computer vision," *Aquacultural Engineering*, vol. 38, no. 2, pp. 97–104, 2008.

[82] ——, "Real-time underwater sorting of edible fish species," *Computers and Electronics in Agriculture*, vol. 56, no. 1, pp. 34–45, 3 2007. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0168169906001244

[83] F. Shafait, A. Mian, M. Shortis, B. Ghanem, P. F. Culverhouse, D. Edgington, D. Cline, M. Ravanbakhsh, J. Seager, and E. S. Harvey, "Fish identification from videos captured in uncontrolled underwater environments," *ICES Journal of Marine Science*, vol. 73, pp. 2737–2746, 2016.

[84] M. A. Islam, M. R. Howlader, U. Habiba, R. H. Faisal, and M. M. Rahman, "Indigenous Fish Classification of Bangladesh using Hybrid Features with SVM Classifier," in *5th International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering, IC4ME2 2019*, 2019.

[85] J. S. Lopez-Villa, H. D. Insuasti-Ceballos, S. Molina-Giraldo, A. Alvarez-Meza, and G. Castellanos-Dominguez, "A novel tool for ground truth data generation for video-based object classification," in *2015 20th Symposium on Signal Processing, Images and Computer Vision, STSIVA 2015 - Conference Proceedings*, 2015.

[86] 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," in *Proceedings - 2015 IEEE Winter Conference on Applications of Computer Vision Workshops, WACVW 2015*, 2015, pp. 57–62.

[87] E. Hossain, S. M. Alam, A. A. Ali, and M. A. Amin, "Fish activity tracking and species identification in underwater video," in *2016 5th International Conference on Informatics, Electronics and Vision, ICIEV 2016*, 2016, pp. 62–66.

[88] P. X. Huang, B. J. Boom, and R. B. Fisher, "GMM improves the reject option in hierarchical classification for fish recognition," in *IEEE Winter Conference on Applications of Computer Vision*. IEEE, 3 2014, pp. 371–376. [Online]. Available: https://ieeexplore.ieee.org/document/6836076

[89] V. Mutneja and S. Singh, "Haar-features training parameters analysis in boosting based machine learning for improved face detection," *International Journal of Advanced Technology and Engineering Exploration*, vol. 8, no. 80, pp. 919–931, 2021.

[90] T. Lindeberg, "Scale Invariant Feature Transform," *Scholarpedia*, vol. 7, no. 5, p. 10491, 2012. [Online]. Available: http://www.scholarpedia.org/article/Scale_Invariant_Feature_Transform

[91] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. I, 2005, pp. 886–893.

[92] A. Rova, G. Mori, and L. M. Dill, "One fish, two fish, butterfish, trumpeter: Recognizing fish in underwater video," in *Proceedings of IAPR Conference on Machine Vision Applications, MVA 2007*, 2007, pp. 404–407.

[93] Y. Hu, A. S. Mian, and R. Owens, "Face Recognition Using Sparse Approximated Nearest Points between Image Sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 1992–2004, 2012.

[94] M. M. M. Fouad, H. M. Zawbaa, N. El-Bendary, and A. E. Hassanien, "Automatic Nile Tilapia fish classification approach using machine learning techniques," in *13th International Conference on Hybrid Intelligent Systems, HIS 2013*, 2014, pp. 173–178.

[95] S. O. Ogunlana, O. Olabode, and S. A. A. Oluwadare, "Fish Classification Using Support Vector Machine," *African Journal of Computing & ICT*, vol. 8, no. 2, pp. 75–82, 2015.

[96] G. Wang, J. N. Hwang, K. Williams, F. Wallace, and C. S. Rose, "Shrinking encoding with two-level codebook learning for fine-grained fish recognition," in *Proceedings - 2nd Workshop on Computer Vision for Analysis of Underwater Imagery, CVAUI 2016 - In Conjunction with International Conference on Pattern Recognition, ICPR 2016*, 2017, pp. 31–36.

[97] M. K. Alsmadi, K. B. Omar, S. A. Noah, and I. Almarashdeh, "Fish recognition based on robust features extraction from size and shape measurements using neural network," *Journal of Computer Science*, vol. 6, no. 10, pp. 1088–1094, 2010.

[98] M. K. Alsmadi, K. B. Omar, and S. A. Mohd Noah, "Fish classification based on robust features extraction from color signature using back-propagation classifier," *Journal of Computer Science*, vol. 7, no. 1, pp. 52–58, 2011.

[99] C. Pornpanomchai, B. Lurstwut, P. Leerasakultham, and W. Kitiyanan, "Shape- and texture-based fish image recognition system," *Kasetsart Journal - Natural Science*, vol. 47, no. 4, pp. 624–634, 2013.

[100] U. A. Badawi and M. K. Alsmadi, "A general fish classification methodology using meta-heuristic algorithm with back propagation classifier," *Journal of Theoretical and Applied Information Technology*, vol. 66, no. 3, pp. 803–812, 2014.

[101] M. Boudhane, B. Nsiri, and H. Toulni, "Optical fish classification using statistics of parts," *International Journal of Mathematics and Computers in Simulation*, vol. 10, pp. 18–22, 2016.

[102] M. S. Nery, A. M. Machado, M. F. Campos, F. L. Pádua, R. Carceroni, and J. P. Queiroz-Neto, "Determining the appropriate feature set for fish classification tasks," in *Brazilian Symposium of Computer Graphic and Image Processing*, vol. 2005, 2005, pp. 173–180.

[103] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[104] D. J. Lee, S. Redd, R. Schoenberger, X. Xu, and P. Zhan, "An Automated Fish Species Classification and Migration Monitoring System," in *IECON Proceedings (Industrial Electronics Conference)*, vol. 2, 2003, pp. 1080–1085.

[105] C. Spampinato, D. Giordano, R. Di Salvo, Y.-H. J. Chen-Burger, R. B. Fisher, and G. Nadarajan, "Automatic fish classification for underwater species behavior understanding," in *Proc. 1st Int. Worksh. Anal. Retriev. Tracked Events Motion Imagery Streams*. Firenze, Italy: ACM, 10 2010, pp. 45–50.

[106] Y. H. Hsiao, C. C. Chen, S. I. Lin, and F. P. Lin, "Real-world underwater fish recognition and identification, using sparse representation," *Ecological Informatics*, vol. 23, pp. 13–21, 2014.

[107] B. J. Boom, J. He, S. Palazzo, P. X. Huang, C. Beyan, H.-M. Chou, F.-P. Lin, C. Spampinato, and R. B. Fisher, "A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage," *Ecological Informatics*, vol. 23, pp. 83–97, 2014.

[108] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT Press, 2016, vol. 1.

[109] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 5 2015. [Online]. Available: http://www.nature.com/articles/nature14539

[110] M. Jahanbakht, W. Xiang, and M. R. Azghadi, "Sea Surface Temperature Forecasting With Ensemble of Stacked Deep Neural Networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9507522/

[111] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," 2007.

[112] T. R. Cook, "Neural Networks," in *Advanced Studies in Theoretical and Applied Econometrics*, 2020, pp. 161–189. [Online]. Available: http://link.springer.com/10.1007/978-3-030-31150-6_6

[113] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts," in *International Conference on Learning Representations*, 2017.

[114] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, p. 85117, 1 2015. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2014.09.003

[115] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, and others, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[116] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition." in *Interspeech*, vol. 11, 2013, pp. 73–75.

[117] K. Korekado, T. Morie, O. Nomura, H. Ando, T. Nakano, M. Matsugu, and A. Iwata, "A convolutional neural network VLSI for image recognition using merged/mixed analog-digital architecture," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer, 2003, pp. 169–176.

[118] R. Proud, R. Mangeni-Sande, R. J. Kayanda, M. J. Cox, C. Nyamweya, C. Ongore, V. Natugonza, I. Everson, M. Elison, L. Hobbs, B. B. Kashindye, E. W. Mlaponi, A. Taabu-Munyaho, V. M. Mwainge, E. Kagoya, A. Pegado, E. Nduwayesu, and A. S. Brierley, "Applications of machine learning and artificial intelligence in marine science: all articles," *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1267–1455, 7 2020. [Online]. Available: https://academic.oup.com/icesjms/article/77/4/1267/5873749

[119] Y. C. Lu, C. Tung, and Y. F. Kuo, "Identifying the species of harvested tuna and billfish using deep convolutional neural networks," *ICES Journal of*

Marine Science, vol. 77, no. 4, pp. 1318–1329, 7 2020. [Online]. Available: https://academic.oup.com/icesjms/article/77/4/1318/5509966

[120] G. French, M. Mackiewicz, M. Fisher, H. Holah, R. Kilburn, N. Campbell, and C. Needle, "Deep neural networks for analysis of fisheries surveillance video and automated monitoring of fish discards," *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1340–1353, 7 2020. [Online]. Available: https://academic.oup.com/icesjms/article/77/4/1340/5542623

[121] C. Liu, S. Zhou, Y. G. Wang, and Z. Hu, "Natural mortality estimation using tree-based ensemble learning models," *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1414–1426, 7 2020. [Online]. Available: https://academic.oup.com/icesjms/article/77/4/1414/5854079

[122] L. Chen, Y. Xia, D. Pan, and C. Wang, "Deep learning based active monitoring for anti-collision between vessels and bridges," in *IABSE Symposium, Guimaraes 2019: Towards a Resilient Built Environment Risk and Asset Management - Report*, 2019.

[123] C. Juliani and E. Juliani, "Deep learning of terrain morphology and pattern discovery via network-based representational similarity analysis for deep-sea mineral exploration," *Ore Geology Reviews*, 2021.

[124] R. Garcia, R. Prados, J. Quintana, A. Tempelaar, N. Gracias, S. Rosen, H. Vågstøl, K. Løvall, H. Vagstol, and K. Lovall, "Automatic segmentation of fish using deep learning with application to fish size measurement," *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1354–1366, 11 2021. [Online]. Available: https://doi.org/10.1093/icesjms/fsz186

[125] N. F. F. Alshdaifat, A. Z. Talib, and M. A. Osman, "Improved deep learning framework for fish segmentation in underwater videos," *Ecological Informatics*, vol. 59, p. 101121, 2020.

[126] W. Zhang, C. Wu, and Z. Bao, "DPANet: Dual Poolingaggregated Attention Network for fish segmentation," *IET Computer Vision*, vol. 16, no. 1, pp. 67–82, 2 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1049/cvi2.12065

[127] Y. Su, L. Guo, Z. Jin, and X. Fu, "A mobile-beacon based iterative localization mechanism in large-scale underwater acoustic sensor networks," *IEEE Internet Things J.*, 9 2020.

[128] A. Jalal, A. Salman, A. Mian, M. Shortis, and F. Shafait, "Fish detection and species classification in underwater environments using deep learning with temporal information," *Ecological Informatics*, vol. 57, p. 101088, 2020.

[129] K. M. Knausgård, A. Wiklund, T. K. Sørdalen, K. T. Halvorsen, A. R. Kleiven, L. Jiao, and M. Goodwin, "Temperate fish detection and classification: a deep learning based approach," *Applied Intelligence*, 2021.

[130] P. Tarling, M. Cantor, A. Clapés, and S. Escalera, "DEEP LEARNING WITH SELF-SUPERVISION AND UNCERTAINTY REGULARIZATION TO COUNT FISH IN UNDERWATER IMAGES," Other, Tech. Rep., 2021. [Online]. Available: http://www.echoview.com.

[131] S. Schneider and A. Zhuang, "Counting Fish and Dolphins in Sonar Images Using Deep Learning," *arXiv preprint arXiv:2007.12808*, 7 2020. [Online]. Available: http://arxiv.org/abs/2007.12808

[132] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 7 2019. [Online]. Available: https://link.springer.com/10.1007/s10618-019-00619-1

[133] B. J. Boom, P. X. Huang, C. Beyan, C. Spampinato, S. Palazzo, J. He, E. Beauxis-Aussalet, S. I. Lin, H. M. Chou, G. Nadarajan, Y. H. Chen-Burger, J. van Ossenbruggen, D. Giordano, L. Hardman, F. P. Lin, and R. B. Fisher, "Long-term underwater camera surveillance for monitoring and analysis of fish populations," *Workshop on Visual observation and Analysis of Animal and Insect Behavior (VAIB), in conjunction with ICPR 2012*, 2012.

[134] Y. Takada, K. Koyama, and T. Usami, "Position Estimation of Small Robotic Fish Based on Camera Information and Gyro Sensors," *Robotics*, vol. 3, no. 2, pp. 149–162, 4 2014. [Online]. Available: http://www.mdpi.com/2218-6581/3/2/149

[135] J. R. Martinez-de Dios, C. Serna, and A. Ollero, "Computer vision and robotics techniques in fish farms," *Robotica*, vol. 21, no. 3, pp. 233–243, 6 2003. [Online]. Available: https://www.cambridge.org/core/product/identifier/S0263574702004733/type/journal_article

[136] M. Boudhane and B. Nsiri, "Underwater image processing method for fish localization and detection in submarine environment," *Journal of Visual Communication and Image Representation*, vol. 39, pp. 226–238, 8 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1047320316300840

[137] E. A. Krupinski, "Receiver operating characteristic (Roc) analysis," *Frontline Learning Research*, vol. 5, no. 3 Special Issue, pp. 31–42, 2017.

[138] A. C. J. Janssens and F. K. Martens, "Reflection on modern methods: Revisiting the area under the ROC Curve," *International Journal of Epidemiology*, vol. 49, no. 4, pp. 1397–1403, 2020.

[139] P. Varalakshmi and J. Julanta Leela Rachel, "Recognition of Fish Categories Using Deep Learning Technique," in *2019 Proceedings of the 3rd International Conference on Computing and Communications Technologies, ICCCT 2019*, 2019.

[140] G. Chen, P. Sun, and Y. Shang, "Automatic fish classification system using deep learning," in *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, 2018.

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

[142] D. Rathi, S. Jain, and S. Indu, "Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning," *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*, 12 2017. [Online]. Available: http://dx.doi.org/10.1109/icapr.2017.8593044

[143] A. B. Tamou, A. Benzinou, K. Nasreddine, and L. Ballihi, "Underwater live fish recognition by deep learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.

[144] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, 2015.

[145] D. A. Konovalov, A. Saleh, M. Bradley, M. Sankupellay, S. Marini, and M. Sheaves, "Underwater Fish Detection with Weak Multi-Domain Supervision,"

in *2019 International Joint Conference on Neural Networks (IJCNN)*, vol. 2019-July. IEEE, 7 2019, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8851907/

[146] P. Zhuang, Y. Wang, and Y. Qiao, "WildFish++: A Comprehensive Fish Benchmark for Multimedia Research," *IEEE Transactions on Multimedia*, 2020.

[147] S. A. Siddiqui, A. Salman, M. I. Malik, F. Shafait, A. Mian, M. R. Shortis, and E. S. Harvey, "Automatic fish species classification in underwater videos: Exploiting pre-trained deep neural network models to compensate for limited labelled data," *ICES Journal of Marine Science*, 2018.

[148] D. A. Pisner and D. M. Schnyer, "Support vector machine," in *Machine Learning: Methods and Applications to Brain Disorders*, 2019.

[149] B. V. Deep and R. Dash, "Underwater Fish Species Recognition Using Deep Learning Techniques," in *2019 6th International Conference on Signal Processing and Integrated Networks, SPIN 2019*, 2019.

[150] M. A. Iqbal, Z. Wang, Z. A. Ali, and S. Riaz, "Automatic Fish Species Classification Using Deep Convolutional Neural Networks," *Wireless Personal Communications*, 2021.

[151] S. Villon, D. Mouillot, M. Chaumont, E. S. Darling, G. Subsol, T. Claverie, and S. Villéger, "A Deep learning method for accurate and fast identification of coral reef fishes in underwater images," *Ecological Informatics*, 2018.

[152] L. Meng, T. Hirayama, and S. Oyanagi, "Underwater-Drone with Panoramic Camera for Automatic Fish Recognition Based on Deep Learning," *IEEE Access*, 2018.

[153] C. Lammie, A. Olsen, T. Carrick, and M. Rahimi Azghadi, "Low-power and high-speed deep FPGA inference engines for weed classification at the edge," *IEEE Access*, 2019.

[154] Y.-H. Hsiao, C.-C. Chen, S.-I. Lin, and F.-P. Lin, "Real-world underwater fish recognition and identification, using sparse representation," *Ecological Informatics*, vol. 23, pp. 13–21, 2014.

[155] J. Kukačka, V. Golkov, and D. Cremers, "Regularization for deep learning: A taxonomy," *arXiv preprint arXiv:1710.10686*, 2017.

[156] M. Zurowietz and T. W. Nattkemper, "Unsupervised Knowledge Transfer for Object Detection in Marine Environmental Monitoring and Exploration," *IEEE Access*, vol. 8, pp. 143 558–143 568, 2020.

[157] C. Qiu, S. Zhang, C. Wang, Z. Yu, H. Zheng, and B. Zheng, "Improving transfer learning and squeeze- and-excitation networks for small-scale fine-grained fish image classification," *IEEE Access*, vol. 6, pp. 78 503–78 512, 2018.

[158] A. Mahmood, M. Bennamoun, S. An, F. Sohel, F. Boussaid, R. Hovey, G. Kendrick, and R. B. Fisher, "Coral classification with hybrid feature representations," in *Proceedings - International Conference on Image Processing, ICIP*, 2016.

[159] 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*, 2016.

[160] J.-N. Blanchet, S. Déry, J.-A. Landry, and K. Osborne, "Automated annotation of corals in natural scene images using multiple texture representations," *PeerJ*, 2016.

[161] I. Nilssen, T. Moller, and T. W. Nattkemper, "Active Learning for the Classification of Species in Underwater Images from a Fixed Observatory," in *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, vol. 2018-Janua, 2017, pp. 2891–2897.

[162] N. Carlevaris-Bianco, A. Mohan, and R. M. Eustice, "Initial results in underwater single image dehazing," in *OCEANS 2010 MTS/IEEE SEATTLE*. IEEE, 9 2010, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/document/5664428/

[163] C. Kumar and P. Prabhaka, "An Image Based Technique for Enhancement of Underwater Images," *Inernational Journal of Machine Intelligence*, vol. 3, no. 4, pp. 217–224, 2011. [Online]. Available: http://arxiv.org/ftp/arxiv/papers/1212/1212.0291.pdf

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

[165] A. Shihavuddin, N. Gracias, R. Garcia, A. Gleason, and B. Gintert, "Image-Based Coral Reef Classification and Thematic Mapping," *Remote Sensing*, vol. 5, no. 4, pp. 1809–1841, 4 2013. [Online]. Available: http://www.mdpi.com/2072-4292/5/4/1809

[166] T. He and X. Li, "Image quality recognition technology based on deep learning," *Journal of Visual Communication and Image Representation*, vol. 65, 2019.

[167] B. G. Van Allen, A. E. Dunham, C. M. Asquith, and V. H. Rudolf, "Life history predicts risk of species decline in a stochastic world," *Proceedings of the Royal Society B: Biological Sciences*, vol. 279, no. 1738, pp. 2691–2697, 2012.

[168] D. F. Shryock, L. A. Defalco, and T. C. Esque, "Life-history traits predict perennial species response to fire in a desert ecosystem," *Ecology and Evolution*, vol. 4, no. 15, pp. 3046–3059, 2014.

[169] S. Vincenzi, A. J. Crivelli, D. Jesesek, E. Campbell, and J. C. Garza, "Effects of species invasion on population dynamics, vital rates and life histories of the native species," *Population Ecology*, vol. 61, no. 1, pp. 25–34, 2019.

[170] S. H. Wang, J. Zhao, X. Liu, Z.-M. Qian, Y. Liu, and Y. Q. Chen, "3D tracking swimming fish school with learned kinematic model using LSTM network," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3 2017, pp. 1068–1072. [Online]. Available: http://ieeexplore.ieee.org/document/7952320/

[171] H. Måløy, A. Aamodt, and E. Misimi, "A spatio-temporal recurrent network for salmon feeding action recognition from underwater videos in aquaculture," *Computers and Electronics in Agriculture*, vol. 167, p. 105087, 12 2019.

[172] Y. Peng, N. Kondo, T. Fujiura, T. Suzuki, Wulandari, H. Yoshioka, and E. Itoyama, "Classification of multiple cattle behavior patterns using a recurrent neural network with long short-term memory and inertial measurement units," *Computers and Electronics in Agriculture*, vol. 157, pp. 247–253, 2 2019.

[173] J.-L. Xu, S. Hugelier, H. Zhu, and A. A. Gowen, "Deep learning for classification of time series spectral images using combined multi-temporal and spectral features," *Analytica Chimica Acta*, vol. 1143, pp. 9–20, 1 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0003267020311429

[174] M. Jahanbakht, W. Xiang, L. Hanzo, and M. R. Azghadi, "Internet of Underwater Things and Big Marine Data Analytics - A Comprehensive Survey," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 904–956, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9328873/

[175] M. Palmer, A. Álvarez-Ellacuría, V. Moltó, and I. A. Catalán, "Automatic, operational, high-resolution monitoring of fish length and catch numbers from landings using deep learning," *Fisheries Research*, vol. 246, p. 106166, 2 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0165783621002940

[176] E. Ditria, S. Lopez-Marcano, M. Sievers, E. Jinks, C. Brown, and R. Connolly, "Automating the analysis of fish abundance using object detection: optimising animal ecology with deep learning," *Frontiers in Marine Science*, 2019.

[177] L. Jing, Y. Chen, and Y. Tian, "Coarse-to-Fine Semantic Segmentation from Image-Level Labels," *IEEE Transactions on Image Processing*, 2020.

[178] D. Pathak, P. Krähenbühl, T. Darrell, P. Krahenbuhl, and T. Darrell, "Constrained Convolutional Neural Networks for Weakly Supervised Segmentation," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1796–1804, 2015.

[179] I. Laradji, P. Rodriguez, O. Manas, K. Lensink, M. Law, L. Kurzman, W. Parker, D. Vazquez, and D. Nowrouzezahrai, "A Weakly Supervised Consistency-based Learning Method for COVID-19 Segmentation in CT Images," in *WACV*, 2021.

[180] X. Qi, Z. Liu, J. Shi, H. Zhao, and J. Jia, "Augmented feedback in semantic segmentation under image level supervision," in *European Conference on Computer Vision*. Springer, 2020, pp. 90–105.

[181] M. C. Chuang, J. N. Hwang, K. Williams, and R. Towler, "Automatic fish segmentation via double local thresholding for trawl-based underwater camera systems," in *Proceedings - International Conference on Image Processing, ICIP*, 2011, pp. 3145–3148.

[182] D. Wang, R. Vinson, M. Holmes, and G. Seibel, "Convolutional neural network guided blue crab knuckle detection for autonomous crab meat picking machine," *Optical Engineering*, 2018.

[183] S. Villon, M. Chaumont, G. Subsol, S. Villéger, T. Claverie, and D. Mouillot, "Coral reef fish detection and recognition in underwater videos by supervised machine learning: Comparison between deep learning and HOG+SVM methods," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.

[184] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of Deep Learning for Object Detection," in *Procedia Computer Science*, 2018.

[185] J. A. Garcia, D. Masip, V. Sbragaglia, and J. Aguzzi, "Automated identification and tracking of nephrops norvegicus (L.) using infrared and monochromatic blue light," in *Frontiers in Artificial Intelligence and Applications*, 2016.

[186] Z. Duan and J. Deng, "Automatic video tracking of chinese mitten crab using particle filter based on multi features," in *2019 IEEE 3rd International Conference on Electronic Information Technology and Computer Engineering, EITCE 2019*, 2019.

[187] D. Kang, Z. Ma, and A. B. Chan, "Beyond counting: comparisons of density maps for crowd analysis tasks-counting, detection, and tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[188] R. Lumauag and M. Nava, "Fish tracking and counting using image processing," in *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management, HNICEM 2018*, 2019.

[189] R. Mandal, R. M. Connolly, T. A. Schlacher, and B. Stantic, "Assessing fish abundance from underwater video using deep neural networks," in *Proceedings of the International Joint Conference on Neural Networks*, 2018.

[190] A. Naseer, E. N. Baro, S. D. Khan, and Y. V. Gordillo, "Automatic Detection of Nephrops norvegicus Burrows in Underwater Images Using Deep Learning," in *2020 Global Conference on Wireless and Optical Technologies, GCWOT 2020*, 2020.

[191] A. Salman, S. A. Siddiqui, F. Shafait, A. Mian, M. R. Shortis, K. Khurshid, A. Ulges, and U. Schwanecke, "Automatic fish detection in underwater videos by a

deep neural network-based hybrid motion learning system," *ICES Journal of Marine Science*, 2020.

[192] S. Zhao, S. Zhang, J. Liu, H. Wang, J. Zhu, D. Li, and R. Zhao, "Application of machine learning in intelligent fish aquaculture: A review," *Aquaculture*, vol. 540, p. 736724, 7 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0044848621003860

[193] L. Yang, Y. Liu, H. Yu, X. Fang, L. Song, D. Li, and Y. Chen, "Computer Vision Models in Intelligent Aquaculture with Emphasis on Fish Detection and Behavior Analysis: A Review," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 2785–2816, 6 2021. [Online]. Available: https://link.springer.com/10.1007/s11831-020-09486-2

[194] Z. Li, W. Li, F. Li, and M. Yuan, "A Review of Computer Vision Technologies for Fish Tracking," *IEEE*, 10 2021. [Online]. Available: http://arxiv.org/abs/2110.02551

[195] M. Moniruzzaman, S. M. S. Islam, M. Bennamoun, and P. Lavery, "Deep learning on underwater marine object detection: A survey," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10617 LNCS. Springer Verlag, 2017, pp. 150–160. [Online]. Available: http://link.springer.com/10.1007/978-3-319-70353-4_13

[196] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A Survey of Optimization Methods From a Machine Learning Perspective," *IEEE Transactions on Cybernetics*, pp. 1–14, 11 2019.

[197] R. Rojas and R. Rojas, "The Backpropagation Algorithm," in *Neural Networks*, 1996.

[198] D. A. Konovalov, A. Saleh, J. A. Domingos, R. D. White, and D. R. Jerry, "Estimating Mass of Harvested Asian Seabass Lates calcarifer from Images," *World Journal of Engineering and Technology*, vol. 6, no. 03, p. 15, 2018.

[199] D. A. Konovalov, A. Saleh, D. B. Efremova, J. A. Domingos, and D. R. Jerry, "Automatic weight estimation of harvested fish from images," in *Digital Image Computing: Techniques and Applications (DICTA)*, 2019, pp. 1–7.

[200] M. Mathur, D. Vasudev, S. Sahoo, D. Jain, and N. Goel, "Crosspooled FishNet: transfer learning based fish species classification model," *Multimedia Tools and Applications*, 2020.

[201] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning Convolutional Neural Networks for Resource Efficient Transfer Learning," *CoRR*, vol. abs/1611.0, 2016. [Online]. Available: http://arxiv.org/abs/1611.06440

[202] K. H. Lee, X. He, L. Zhang, and L. Yang, "CleanNet: Transfer Learning for Scalable Image Classifier Training with Label Noise," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[203] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

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

[205] A. M. Potdar, D. G. Narayan, S. Kengond, and M. M. Mulla, "Performance Evaluation of Docker Container and Virtual Machine," in *Procedia Computer Science*, vol. 171, 2020, pp. 1419–1428.

[206] M. S. Abdul, S. M. Sam, N. Mohamed, K. Kamardin, and R. A. Dziyauddin, "Docker Containers Usage in the Internet of Things: A Survey," *Open International Journal of Informatics (OIJI)*, vol. 7, no. 2, pp. 208–220, 2019. [Online]. Available: http://apps.razak.utm.my/ojs/index.php/oiji/article/view/233

[207] K. Khazukov, V. Shepelev, T. Karpeta, S. Shabiev, I. Slobodin, I. Charbadze, and I. Alferova, "Real-time monitoring of traffic parameters," *Journal of Big Data*, vol. 7, no. 1, 12 2020.

[208] S. Zhang, G. Wu, J. P. Costeira, and J. M. F. Moura, "Understanding Traffic Density from Large-Scale Web Camera Data," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5898–5907.

[209] E. M. Ditria, S. Lopez-Marcano, M. Sievers, E. L. Jinks, C. J. Brown, and R. M. Connolly, "Automating the Analysis of Fish Abundance Using Object Detection: Optimizing Animal Ecology With Deep Learning," *Frontiers in Marine Science*,

vol. 7, 6 2020. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmars.2020.00429/full

[210] E. M. Ditria, M. Sievers, S. Lopez-Marcano, E. L. Jinks, and R. M. Connolly, "Deep learning for automated analysis of fish abundance: the benefits of training across multiple habitats," *Environmental Monitoring and Assessment*, 2020.

[211] L. Liu, H. Lu, Z. Cao, and Y. Xiao, "Counting Fish in Sonar Images," in *Proceedings - International Conference on Image Processing, ICIP*, 2018.

[212] Y. Xue, N. Ray, J. Hugh, and G. Bigras, "Cell counting by regression using convolutional neural network," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.

[213] S. S. Beauchemin and J. L. Barron, "The Computation of Optical Flow," *ACM Computing Surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 9 1995. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/212094.212141

[214] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, vol. 27, pp. 773–780, 2006.

[215] S. Chaudhari, N. Malkan, A. Momin, and M. Bonde, "Yolo Real Time Object Detection," *International Journal of Computer Trends and Technology*, 2020.

[216] A. Joly, H. Goëau, H. Glotin, C. Spampinato, P. Bonnet, W.-P. Vellinga, R. Planque, A. Rauber, R. Fisher, and H. Müller, "LifeCLEF 2014: Multimedia Life Species Identification Challenges," in *Information Access Evaluation. Multilinguality, Multimodality, and Interaction*, ser. Lecture Notes in Computer Science, E. Kanoulas, M. Lupu, P. Clough, M. Sanderson, M. Hall, A. Hanbury, and E. Toms, Eds., vol. 8685. Cham: Springer International Publishing, 2014, pp. 229–249.

[217] A. Salman, S. A. Siddiqui, F. Shafait, A. Mian, M. R. Shortis, K. Khurshid, A. Ulges, and U. Schwanecke, "Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system," *ICES Journal of Marine Science*, 2019.

[218] D. Giordano, S. Palazzo, and C. Spampinato, "Fish4Knowledge: Collecting and Analyzing Massive Coral Reef Fish Video Data," *Intelligent Systems Reference Library*, 2016.

[219] W. Xu and S. Matzner, "Underwater Fish Detection Using Deep Learning for Water Power Applications," in *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 12 2018, pp. 313–318. [Online]. Available: https://ieeexplore.ieee.org/document/8947884/

[220] S. Choi, "Fish identification in underwater video with deep convolutional neural network," CLEF, Tech. Rep., 2015. [Online]. Available: http://ceur-ws.org/Vol-1391/110-CR.pdf

[221] 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.

[222] A. B. Labao and P. C. Naval, "Cascaded deep network systems with linked ensemble components for underwater fish detection in the wild," *Ecological Informatics*, 2019.

[223] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[224] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[225] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *CoRR*, vol. abs/1512.0, 2015.

[226] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[227] P. Zhuang, L. Xing, Y. Liu, S. Guo, and Y. Qiao, "Marine Animal detection and Recognition with advanced deep learning models," in *CEUR Workshop Proceedings*, 2017.

[228] F. Han, J. Yao, H. Zhu, and C. Wang, "Marine Organism Detection and Classification from Underwater Vision Based on the Deep CNN Method," *Mathematical Problems in Engineering*, 2020.

[229] X. Li, M. Shang, H. Qin, and L. Chen, "Fast accurate fish detection and recognition of underwater images with Fast R-CNN," in *OCEANS 2015 - MTS/IEEE Washington*, 2015, pp. 1–5.

[230] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[231] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[232] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation." *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 640–651, 4 2017. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/27244717

[233] A. B. Labao and P. C. Naval, "Simultaneous Localization and Segmentation of Fish Objects Using Multi-task CNN and Dense CRF," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019.

[234] Z.-M. Qian, S. H. Wang, X. E. Cheng, and Y. Q. Chen, "An effective and robust method for tracking multiple fish in video image based on fish head detection," *BMC Bioinformatics*, vol. 17, no. 1, p. 251, 12 2016. [Online]. Available: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1138-y

[235] S. Wang and P. Kanwar, "BFloat16: The secret to high performance on Cloud TPUs {$\vert$} Google Cloud Blog," 6 2021. [Online]. Available: https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus

[236] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 6 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7485869/

[237] A. B. Labao and P. C. Naval, "Weakly-Labelled semantic segmentation of fish objects in underwater videos using a deep residual network," in *Lecture Notes in*

Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2017.

[238] M. Rajchl, M. C. H. Lee, O. Oktay, K. Kamnitsas, J. Passerat-Palmbach, W. Bai, M. Damodaram, M. A. Rutherford, J. V. Hajnal, and B. Kainz, "Deepcut: Object segmentation from bounding box annotations using convolutional neural networks," *IEEE transactions on medical imaging*, vol. 36, no. 2, p. 683, 2016.

[239] A. Khoreva, R. Benenson, J. H. Hosang, M. Hein, and B. Schiele, "Simple Does It: Weakly Supervised Instance and Semantic Segmentation." *CVPR*, pp. 876–885, 2017.

[240] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *ICCV*, 2015, pp. 1635–1643.

[241] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," in *IEEE*, 2016, pp. 3159–3167.

[242] A. L. Bearman, O. Russakovsky, V. Ferrari, L. Fei-Fei, A. L. Bearman, V. Ferrari, F.-F. Li, O. Russakovsky, V. Ferrari, and L. Fei-Fei, "What's the point: Semantic segmentation with point supervision," *ECCV*, vol. abs/1506.0, 2016.

[243] J. Ahn and S. Kwak, "Learning Pixel-Level Semantic Affinity with Image-Level Supervision for Weakly Supervised Semantic Segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[244] Z. Huang, J. XinggangWang, W. Liu, and J. Wang, "Weakly-supervised semantic segmentation network with deep seeded region growing," in *IEEE*, 2018, pp. 7014–7023.

[245] Y. Wei, H. Xiao, H. Shi, Z. Jie, J. Feng, and T. S. Huang, "Revisiting dilated convolution: A simple approach for weakly-and semi-supervised semantic segmentation," in *IEEE*, 2018, pp. 7268–7277.

[246] M. Mancusi, N. Zonca, E. Rodolà, and S. Zuffi, "Fish sounds: towards the evaluation of marine acoustic biodiversity through data-driven audio source separation," *IEEE*, 1 2022. [Online]. Available: https://arxiv.org/abs/2201.05013v2

[247] L. Muñoz, E. Aspillaga, M. Palmer, J. L. Saraiva, and P. Arechavala-Lopez, "Acoustic Telemetry: A Tool to Monitor Fish Swimming Behavior in Sea-Cage Aquaculture," *Frontiers in Marine Science*, vol. 7, p. 645, 7 2020.

[248] K. J. Benoit-Bird and G. L. Lawson, "Ecological Insights from Pelagic Habitats Acquired Using Active Acoustic Techniques," *Annual Review of Marine Science*, vol. 8, pp. 463–490, 1 2016.

[249] E. McCann, L. Li, K. Pangle, N. Johnson, and J. Eickholt, "An underwater observation dataset for fish classification and fishery assessment," *Scientific Data 2018 5:1*, vol. 5, no. 1, pp. 1–8, 10 2018. [Online]. Available: https://www.nature.com/articles/sdata2018190https://www.nature.com/articles/sdata2018190/

[250] X. Zhou, C. Yu, S. Yuan, X. Yuan, H. Yu, and C. Luo, "Learning Visual Representation of Underwater Acoustic Imagery Using Transformer-Based Style Transfer Method," *IEEE*, 11 2022. [Online]. Available: https://arxiv.org/abs/2211.05396v1

[251] A. Ben Tamou, A. Benzinou, and K. Nasreddine, "Targeted Data Augmentation and Hierarchical Classification with Deep Learning for Fish Species Identification in Underwater Images," *Journal of imaging*, vol. 8, no. 8, 8 2022. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/36005457/

[252] G. Fu and Y. Yuna, "Phenotyping and phenomics in aquaculture breeding," *Aquaculture and Fisheries*, vol. 7, no. 2, pp. 140–146, 3 2022.

[253] R. J. Lennox, K. Aarestrup, S. J. Cooke, P. D. Cowley, Z. D. Deng, A. T. Fisk, R. G. Harcourt, M. Heupel, S. G. Hinch, K. N. Holland, N. E. Hussey, S. J. Iverson, S. T. Kessel, J. F. Kocik, M. C. Lucas, J. M. Flemming, V. M. Nguyen, M. J. Stokesbury, S. Vagle, D. L. Vanderzwaag, F. G. Whoriskey, and N. Young, "Envisioning the Future of Aquatic Animal Tracking: Technology, Science, and Application," 2017.

[254] X. Zhao, S. Yan, and Q. Gao, "An Algorithm for Tracking Multiple Fish Based on Biological Water Quality Monitoring," *IEEE Access*, 2019.

[255] M. M. Saberioon and P. Cisar, "Automated multiple fish tracking in three-Dimension using a Structured Light Sensor," *Computers and Electronics in Agriculture*, 2016.

[256] Y. Zhou, H. Yu, J. Wu, Z. Cui, and F. Zhang, "Fish Behavior Analysis Based on Computer Vision: A Survey," *Communications in Computer and Information Science*, vol. 1059, pp. 130–141, 2019. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-15-0121-0_10

[257] D. Li, G. Wang, L. Du, Y. Zheng, and Z. Wang, "Recent advances in intelligent recognition methods for fish stress behavior," *Aquacultural Engineering*, vol. 96, p. 102222, 2 2022.

[258] B. Niu, G. Li, F. Peng, J. Wu, L. Zhang, and Z. Li, "Survey of Fish Behavior Analysis by Computer Vision," *Journal of Aquaculture Research and Development*, vol. 09, no. 05, 2018.

[259] C. W. Fu, J. L. Horng, and M. Y. Chou, "Fish Behavior as a Neural Proxy to Reveal Physiological States," *Frontiers in Physiology*, vol. 13, p. 1420, 7 2022.

[260] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*, 2015.

[261] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, 2017, pp. 105–114.

[262] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative Adversarial Networks," *ArXiv*, vol. abs/1406.2, 2014.

[263] L.-J. Li, K. Li, F. F. Li, J. Deng, W. Dong, R. Socher, and L. Fei-Fei, "ImageNet: a Large-Scale Hierarchical Image Database Shrimp Project View project hybrid intrusion detction systems View project ImageNet: A Large-Scale Hierarchical Image Database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[264] 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 *Lecture Notes in*

Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014.

[265] B. D. De Vos, J. M. Wolterink, P. A. De Jong, T. Leiner, M. A. Viergever, and I. Isgum, "ConvNet-Based Localization of Anatomical Structures in 3-D Medical Images," *IEEE Transactions on Medical Imaging*, 2017.

[266] S. Wörz and K. Rohr, "Localization of anatomical point landmarks in 3D medical images by fitting 3D parametric intensity models," *Medical Image Analysis*, 2006.

[267] Z. H. Zhou, "A brief introduction to weakly supervised learning," 2018.

[268] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free? - Weakly-supervised learning with convolutional neural networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.

[269] I. Laradji, A. Saleh, P. Rodriguez, D. Nowrouzezahrai, M. R. Azghadi, and D. Vazquez, "Affinity LCFCN: Learning to Segment Fish with Weak Supervision," *Scientific Reports*, 11 2020. [Online]. Available: http://arxiv.org/abs/2011.03149

[270] K. L. Zhao, X. L. Jin, and Y. Z. Wang, "Survey on Few-shot Learning," 2021.

[271] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a Few Examples," *ACM Computing Surveys*, vol. 53, no. 3, pp. 1–34, 5 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3386252

[272] S. Villon, C. Iovan, M. Mangeas, and L. Vigliola, "Confronting Deep-Learning and Biodiversity Challenges for Automatic Video-Monitoring of Marine Ecosystems," *Sensors*, vol. 22, no. 2, p. 497, 1 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/2/497

[273] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-Scale Long-Tailed Recognition in an Open World," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2019-June. IEEE, 6 2019, pp. 2532–2541. [Online]. Available: https://ieeexplore.ieee.org/document/8953407/

[274] S. Villon, C. Iovan, M. Mangeas, T. Claverie, D. Mouillot, S. Villéger, and L. Vigliola, "Automatic underwater fish species classification with limited data

using few-shot learning," *Ecological Informatics*, vol. 63, p. 101320, 7 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1574954121001114

[275] J. Feng and X. Xiao, "Multiobject Tracking of Wildlife in Videos Using Few-Shot Learning," *Animals*, vol. 12, no. 9, p. 1223, 5 2022. [Online]. Available: https://www.mdpi.com/2076-2615/12/9/1223

[276] J. Feng and J. Li, "An Adaptive Embedding Network with Spatial Constraints for the Use of Few-Shot Learning in Endangered-Animal Detection," *ISPRS International Journal of Geo-Information*, vol. 11, no. 4, p. 256, 4 2022. [Online]. Available: https://www.mdpi.com/2220-9964/11/4/256

[277] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, vol. 2017-October. IEEE, 10 2017, pp. 2999–3007. [Online]. Available: http://ieeexplore.ieee.org/document/8237586/

[278] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-Balanced Loss Based on Effective Number of Samples," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2019-June. IEEE, 6 2019, pp. 9260–9269. [Online]. Available: https://ieeexplore.ieee.org/document/8953804/

[279] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[280] A. Bendale and T. E. Boult, "Towards Open Set Deep Networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2016-December. IEEE, 6 2016, pp. 1563–1572. [Online]. Available: http://ieeexplore.ieee.org/document/7780542/

[281] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[282] A. Amadori, "Distilling knowledge from Neural Networks to build smaller and faster models," 2019.

[283] P. Wang, Q. Chen, X. He, and J. Cheng, "Towards Accurate Post-training Network Quantization via Bit-Split and Stitching," in *PMLR*. PMLR, 11 2020, pp. 9847–9856. [Online]. Available: http://proceedings.mlr.press/v119/wang20c.html

[284] A. G. Rassadin and A. V. Savchenko, "Compressing deep convolutional neural networks in visual emotion recognition," in *CEUR Workshop Proceedings*, 2017.

[285] R. K. Kushawaha, S. Kumar, B. Banerjee, and R. Velmurugan, "Distilling Spikes: Knowledge Distillation in Spiking Neural Networks," in *IEEE*, 2021.

[286] T. Shimada, H. Bao, I. Sato, and M. Sugiyama, "Classification From Pairwise Similarities/Dissimilarities and Unlabeled Data via Empirical Risk Minimization," *Neural Computation*, vol. 33, no. 5, pp. 1234–1268, 4 2021. [Online]. Available: https://direct.mit.edu/neco/article/33/5/1234/97483/Classification-From-Pairwise-Similarities

[287] H. Wu and S. Prasad, "Semi-Supervised Deep Learning Using Pseudo Labels for Hyperspectral Image Classification," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1259–1270, 3 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8105856/

[288] R. Garcia, R. Prados, J. Quintana, A. Tempelaar, N. Gracias, S. Rosen, H. Vagstol, and K. Lovall, "Automatic segmentation of fish using deep learning with application to fish size measurement," *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1354–1366, 7 2020. [Online]. Available: https://academic.oup.com/icesjms/article/77/4/1354/5602457

[289] C. C. Chang, Y. P. Wang, and S. C. Cheng, "Fish segmentation in sonar images by mask r-cnn on feature maps of conditional random fields," *Sensors*, 2021.

[290] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *IEEE*, 10 2020. [Online]. Available: http://arxiv.org/abs/2010.11929

[291] K. Xu, L. Wen, G. Li, and Q. Huang, "Self-Supervised Deep TripleNet for Video Object Segmentation," *IEEE Transactions on Multimedia*, 2021.

[292] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.

[293] I. Croitoru, S. V. Bogolin, and M. Leordeanu, "Unsupervised Learning of Foreground Object Segmentation," *International Journal of Computer Vision*, 2019.

[294] D. Wei, J. Lim, A. Zisserman, and W. T. Freeman, "Learning and Using the Arrow of Time," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[295] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019.

[296] J. Shao, X. Wen, B. Zhao, and X. Xue, "Temporal Context Aggregation for Video Retrieval with Contrastive Learning," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 1 2021, pp. 3267–3277. [Online]. Available: https://ieeexplore.ieee.org/document/9423060/

[297] P. O. Pinheiro, A. Almahairi, R. Y. Benmalek, F. Golemo, and A. Courville, "Unsupervised Learning of Dense Visual Representations," *Advances in Neural Information Processing Systems*, 11 2020. [Online]. Available: http://arxiv.org/abs/2011.05499

[298] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," *Technologies*, 2020.

[299] M. Ye, X. Zhang, P. C. Yuen, and S.-F. Chang, "Unsupervised Embedding Learning via Invariant and Spreading Instance Feature," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 6 2019, pp. 6203–6212. [Online]. Available: https://ieeexplore.ieee.org/document/8953747/

[300] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[301] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum Contrast for Unsupervised Visual Representation Learning," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.

[302] Z. Lai and W. Xie, "Self-supervised learning for video correspondence flow," in *30th British Machine Vision Conference 2019, BMVC 2019*, 2020.

[303] N. Wang, W. Zhou, and H. Li, "Contrastive Transformation for Self-supervised Correspondence Learning," *IEEE*, 12 2020. [Online]. Available: https://arxiv.org/abs/2012.05057v1

[304] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[305] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," *IEEE*, 7 2016. [Online]. Available: https://arxiv.org/abs/1607.06450v1

[306] N. Wang, W. Zhou, and H. Li, "Contrastive Transformation for Self-supervised Correspondence Learning," *IEEE*, 12 2020. [Online]. Available: https://arxiv.org/abs/2012.05057v1

[307] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 6 2018, pp. 8934–8943. [Online]. Available: https://ieeexplore.ieee.org/document/8579029/

[308] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019.

[309] L. Chi, B. Jiang, and Y. Mu, "Fast Fourier Convolution," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 4479–4488. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/2fd5d41ec6cfab47e32164d5624269b1-Paper.pdf

[310] R. Strudel, R. Garcia, I. Laptev Inria, and C. Schmid Inria, "Segmenter: Transformer for Semantic Segmentation," *IEEE*, 5 2021. [Online]. Available: https://arxiv.org/abs/2105.05633v3

[311] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan, "MetaFormer is Actually What You Need for Vision," *IEEE*, 11 2021. [Online]. Available: https://arxiv.org/abs/2111.11418v2

[312] A. El-Nouby, H. Touvron, M. Caron, P. Bojanowski, M. Douze, A. Joulin, I. Laptev, N. Neverova, G. Synnaeve, J. Verbeek, and H. Jegou, "XCiT: Cross-Covariance Image Transformers," *IEEE*, 6 2021. [Online]. Available: https://arxiv.org/abs/2106.09681v2

[313] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," *37th International Conference on Machine Learning, ICML 2020*, 2 2020. [Online]. Available: http://arxiv.org/abs/2002.05709

[314] L. Zou, M. Zhao, F. Cao, S. Zan, X. Cheng, and X. Liu, "Fish Tracking Based on Feature Fusion and Scale Adaptation in a Real-World Underwater Environment," *Marine Technology Society Journal*, vol. 55, no. 2, pp. 45–53, 3 2021. [Online]. Available: https://www.ingentaconnect.com/content/10.4031/MTSJ.55.2.12

[315] P. Gatti, J. A. D. Fisher, F. Cyr, P. S. Galbraith, D. Robert, and A. Le Bris, "A review and tests of validation and sensitivity of geolocation models for marine fish tracking," *Fish and Fisheries*, vol. 22, no. 5, pp. 1041–1066, 9 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/faf.12568

[316] Y. Wageeh, H. E.-D. Mohamed, A. Fadl, O. Anas, N. ElMasry, A. Nabil, and A. Atia, "YOLO fish detection with Euclidean tracking in fish farms," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 5–12, 1 2021. [Online]. Available: http://link.springer.com/10.1007/s12652-020-02847-6

[317] V. G. Guida, P. C. Valentine, and L. B. Gallea, "Semidiurnal Temperature Changes Caused by Tidal Front Movements in the Warm Season in Seabed Habitats on the Georges Bank Northern Margin and Their Ecological Implications," *PLoS ONE*, vol. 8, no. 2, p. e55273, 2 2013. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0055273

[318] J. Sundin, R. Morgan, M. H. Finnøen, A. Dey, K. Sarkar, and F. Jutfelt, "On the Observation of Wild Zebrafish (Danio rerio) in India," *Zebrafish*, vol. 16, no. 6, pp. 546–553, 12 2019. [Online]. Available: https://www.liebertpub.com/doi/10.1089/zeb.2019.1778

[319] E. M. Olsen, M. R. Heupel, C. A. Simpfendorfer, and E. Moland, "Harvest selection on Atlantic cod behavioral traits: implications for spatial management," *Ecology and Evolution*, vol. 2, no. 7, pp. 1549–1562, 7 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/ece3.244

[320] N. X. R. Wang, S. Cullis-Suzuki, and A. Branzan Albu, "Automated Analysis of Wild Fish Behavior in a Natural Habitat," in *Proceedings of the 2nd International Workshop on Environmental Multimedia Retrieval*. New York, NY, USA: ACM, 6 2015, pp. 21–26. [Online]. Available: https://dl.acm.org/doi/10.1145/2764873.2764875

[321] D. A. Konovalov, A. Saleh, D. B. Efremova, J. A. Domingos, and D. R. Jerry, "Automatic Weight Estimation of Harvested Fish from Images," in *2019 Digital Image Computing: Techniques and Applications, DICTA 2019*. Institute of Electrical and Electronics Engineers Inc., 12 2019.

[322] R. Yao, G. Lin, S. Xia, J. Zhao, and Y. Zhou, "Video Object Segmentation and Tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 4, pp. 1–47, 8 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3391743

[323] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele, "Lucid Data Dreaming for Video Object Segmentation," *International Journal of Computer Vision*, vol. 127, no. 9, pp. 1175–1197, 9 2019. [Online]. Available: http://link.springer.com/10.1007/s11263-019-01164-6

[324] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixe, D. Cremers, and L. Van Gool, "Video Object Segmentation without Temporal Information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 6, pp. 1515–1530, 6 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8362936/

[325] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, "Deep neural network concepts for background subtraction:A systematic review and comparative evaluation," *Neural Networks*, vol. 117, pp. 8–66, 9 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0893608019301303

[326] R. Kalsotra and S. Arora, "A Comprehensive Survey of Video Datasets for Background Subtraction," *IEEE Access*, vol. 7, pp. 59 143–59 171, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8706931/

[327] B. Garcia-Garcia, T. Bouwmans, and A. J. Rosales Silva, "Background subtraction in real applications: Challenges, current models and future directions," *Computer Science Review*, vol. 35, p. 100204, 2 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1574013718303101

[328] H. Pan, G. Zhu, C. Peng, and Q. Xiao, "Background subtraction for night videos," *PeerJ Computer Science*, vol. 7, p. e592, 6 2021. [Online]. Available: https://peerj.com/articles/cs-592

[329] L. Maddalena and A. Petrosino, "Background Subtraction for Moving Object Detection in RGBD Data: A Survey," *Journal of Imaging*, vol. 4, no. 5, p. 71, 5 2018. [Online]. Available: http://www.mdpi.com/2313-433X/4/5/71

[330] S. Lu, Z. Luo, F. Gao, M. Liu, K. Chang, and C. Piao, "A Fast and Robust Lane Detection Method Based on Semantic Segmentation and Optical Flow Estimation," *Sensors*, vol. 21, no. 2, p. 400, 1 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/2/400

[331] S. Anthwal and D. Ganotra, "An overview of optical flow-based approaches for motion segmentation," *The Imaging Science Journal*, vol. 67, no. 5, pp. 284–294, 7 2019. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/13682199.2019.1641316

[332] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang, "SegFlow: Joint Learning for Video Object Segmentation and Optical Flow," in *2017 IEEE International Conference on Computer Vision (ICCV)*, vol. 2017-October. IEEE, 10 2017, pp. 686–695. [Online]. Available: http://ieeexplore.ieee.org/document/8237343/

[333] M. Ding, Z. Wang, B. Zhou, J. Shi, Z. Lu, and P. Luo, "Every Frame Counts: Joint Learning of Video Segmentation and Optical Flow," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 10 713–10 720, 4 2020. [Online]. Available: https://aaai.org/ojs/index.php/AAAI/article/view/6699

[334] A. Garcia-Dopico, J. L. Pedraza, M. Nieto, A. Pérez, S. Rodríguez, and L. Osendi, "Locating moving objects in car-driving sequences," *EURASIP Journal on Image and Video Processing*, vol. 2014, no. 1, p. 24, 12 2014. [Online]. Available: https://jivp-eurasipjournals.springeropen.com/articles/10.1186/1687-5281-2014-24

[335] S. Chraa Mesbahi, M. A. Mahraz, J. Riffi, and H. Tairi, "Head Gesture Recognition Using Optical Flow Based Background Subtraction," in *Lecture Notes in Networks and Systems*, 2018, vol. 37, pp. 200–211. [Online]. Available: http://link.springer.com/10.1007/978-3-319-74500-8_18

[336] A. Kushwaha, A. Khare, O. Prakash, and M. Khare, "Dense optical flow based background subtraction technique for object segmentation in moving camera environment," *IET Image Processing*, vol. 14, no. 14, pp. 3393–3404, 12 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1049/iet-ipr.2019.0960

[337] D. Sun, C. Liu, and H. Pfister, "Local layering for joint motion estimation and occlusion detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[338] Z. Chen, H. Jin, Z. Lin, S. Cohen, and Y. Wu, "Large displacement optical flow from nearest neighbor fields," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013.

[339] T. Brox and J. Malik, "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 500–513, 3 2011. [Online]. Available: http://ieeexplore.ieee.org/document/5551149/

[340] H. Guan, X. Y. Xue, and Z. Y. An, "Advances on application of deep learning for video object tracking," 2016.

[341] G. Ciaparrone, F. Luque Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, "Deep learning in video multi-object tracking: A survey," *Neurocomputing*, vol. 381, pp. 61–88, 3 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231219315966

[342] R. Gomez-Nieto, J. F. Ruiz-Munoz, J. Beron, C. A. A. Franco, H. D. Benitez-Restrepo, and A. C. Bovik, "Quality Aware Features for Performance Prediction and Time Reduction in Video Object Tracking," *IEEE Access*, vol. 10, pp. 13 290–13 310, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9698081/

[343] J. Qiu, L. Wang, Y. H. Hu, and Y. Wang, "Two motion models for improving video object tracking performance," *Computer Vision and Image*

Understanding, vol. 195, p. 102951, 6 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1077314220300345

[344] X. Kang, B. Song, and F. Sun, "A Deep Similarity Metric Method Based on Incomplete Data for Traffic Anomaly Detection in IoT," *Applied Sciences*, vol. 9, no. 1, p. 135, 1 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/1/135

[345] A. Dadgar, Y. Baleghi, and M. Ezoji, "Improved Object Matching in Multi-Objects Tracking Based On Zernike Moments and Combination of Multiple Similarity Metrics," *International Journal of Engineering*, vol. 34, no. 6, 6 2021. [Online]. Available: http://www.ije.ir/article_130893.html

[346] S. Bag, S. K. Kumar, and M. K. Tiwari, "An efficient recommendation generation using relevant Jaccard similarity," *Information Sciences*, vol. 483, pp. 53–64, 5 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0020025519300325

[347] B. Zhu, Y. Jiang, M. Gu, and Y. Deng, "A GPU Acceleration Framework for Motif and Discord Based Pattern Mining," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 8, pp. 1987–2004, 8 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9343677/

[348] J. Zhu, Z. Wang, S. Wang, and S. Chen, "Moving Object Detection Based on Background Compensation and Deep Learning," *Symmetry*, vol. 12, no. 12, p. 1965, 11 2020. [Online]. Available: https://www.mdpi.com/2073-8994/12/12/1965

[349] M.-N. Chapel and T. Bouwmans, "Moving objects detection with a moving camera: A comprehensive review," *Computer Science Review*, vol. 38, p. 100310, 11 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1574013720304106X

[350] H. Zhu, H. Wei, B. Li, X. Yuan, and N. Kehtarnavaz, "A Review of Video Object Detection: Datasets, Metrics and Methods," *Applied Sciences*, vol. 10, no. 21, p. 7834, 11 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/21/7834

[351] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A Survey of Deep Learning-Based Object Detection," *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8825470/

[352] 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, 11 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8627998/

[353] T. Jiang, J. L. Gradus, and A. J. Rosellini, "Supervised Machine Learning: A Brief Primer," *Behavior Therapy*, 2020.

[354] X. Wang, X. Lin, and X. Dang, "Supervised learning in spiking neural networks: A review of algorithms and evaluations," *Neural Networks*, 2020.

[355] Z. Zhou, R. Zhang, and D. Yin, "A strong feature representation for siamese network tracker," *Multimedia Tools and Applications*, vol. 79, no. 35-36, pp. 25 873–25 887, 9 2020. [Online]. Available: https://link.springer.com/10.1007/s11042-020-09164-2

[356] J. Peng, J. Li, and X. Shang, "A learning-based method for drug-target interaction prediction based on feature representation learning and deep neural network," *BMC Bioinformatics*, vol. 21, no. S13, p. 394, 9 2020. [Online]. Available: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-03677-1

[357] Y. Xie, Z. Du, J. Li, M. Jing, E. Chen, and K. Lu, "Joint metric and feature representation learning for unsupervised domain adaptation," *Knowledge-Based Systems*, vol. 192, p. 105222, 3 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0950705119305489

[358] R. Liu, Z. Wu, S. X. Yu, and S. Lin, "The Emergence of Objectness: Learning Zero-Shot Segmentation from Videos," *Advances in Neural Information Processing Systems*, vol. 16, pp. 13 137–13 152, 11 2021. [Online]. Available: https://arxiv.org/abs/2111.06394v1

[359] N. A. Golilarz, H. Demirel, and H. Gao, "Adaptive Generalized Gaussian Distribution Oriented Thresholding Function for Image De-Noising," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 2, 2019. [Online]. Available: http://thesai.org/Publications/ViewPaper?Volume=10&Issue=2&Code=ijacsa&SerialNo=2

[360] Z. Teed and J. Deng, "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow (Extended Abstract)," in *Proceedings of the Thirtieth International Joint*

Conference on Artificial Intelligence. California: International Joint Conferences on Artificial Intelligence Organization, 8 2021, pp. 4839–4843. [Online]. Available: https://www.ijcai.org/proceedings/2021/662

[361] D. T. Nguyen, M. Dax, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, Z. Lou, and T. Brox, "DeepUSPS: Deep robust unsupervised saliency prediction with self-supervision," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[362] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *PAMI*, vol. 40, no. 4, pp. 834–848, 2018.

[363] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in neural information processing systems*, 2011, pp. 109–117.

[364] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: Dynamic and fast instance segmentation," in *Advances in Neural Information Processing Systems*, vol. 2020-December, 2020.

[365] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "SOLO: Segmenting Objects by Locations," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12363 LNCS, pp. 649–665. [Online]. Available: https://link.springer.com/10.1007/978-3-030-58523-5_38

[366] A. Neubeck and L. Van Gool, "Efficient Non-Maximum Suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3. IEEE, 2006, pp. 850–855. [Online]. Available: http://ieeexplore.ieee.org/document/1699659/

[367] OpenCv, "OpenCV Library," *OpenCV Website*, p. All, 2014. [Online]. Available: https://opencv.org/about.html

[368] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," *Proceedings - International Conference on Image Processing, ICIP*, vol. 2016-August, pp. 3464–3468, 2 2016. [Online]. Available: http://arxiv.org/abs/1602.00763http://dx.doi.org/10.1109/ICIP.2016.7533003

[369] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 3 1960. [Online]. Available: https://asmedigitalcollection.asme.org/fluidsengineering/article/82/1/35/397706/A-New-Approach-to-Linear-Filtering-and-Prediction

[370] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, 1955.

[371] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum, "Learning to Detect a Salient Object," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 353–367, 2 2011. [Online]. Available: http://ieeexplore.ieee.org/document/5432215/

[372] X. Chen, R. Girshick, K. He, and P. Dollar, "TensorMask: A Foundation for Dense Object Segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, vol. 2019-October. IEEE, 10 2019, pp. 2061–2069. [Online]. Available: https://ieeexplore.ieee.org/document/9010024/

[373] X. Chen, P. Zhang, L. Quan, C. Yi, and C. Lu, "Underwater Image Enhancement based on Deep Learning and Image Formation Model," *IEEE*, 1 2021. [Online]. Available: https://arxiv.org/abs/2101.00991v2

[374] Z. Xiao, Y. Han, S. Rahardja, and Y. Ma, "USLN: A statistically guided lightweight network for underwater image enhancement via dual-statistic white balance and multi-color space stretch," *IEEE*, 9 2022. [Online]. Available: https://arxiv.org/abs/2209.02221v1

[375] Z. Fu, W. Wang, Y. Huang, X. Ding, and K.-K. Ma, "Uncertainty Inspired Underwater Image Enhancement," in *AISTATS*, 2022, pp. 465–482. [Online]. Available: https://link.springer.com/10.1007/978-3-031-19797-0_27

[376] A. Galdran, D. Pardo, A. Picón, and A. Alvarez-Gila, "Automatic red-channel underwater image restoration," *Journal of Visual Communication and Image Representation*, vol. 26, pp. 132–145, 2015.

[377] P. Drews, E. Nascimento, F. Moraes, S. Botelho, and M. Campos, "Transmission estimation in underwater single images," in *Proceedings of the IEEE international conference on computer vision workshops*, 2013, pp. 825–830.

[378] W. Song, Y. Wang, D. Huang, and D. Tjondronegoro, "A rapid scene depth estimation model based on underwater light attenuation prior for underwater image restoration," in *Pacific Rim Conference on Multimedia*, 2018, pp. 678–688.

[379] 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.

[380] D. Berman, T. Treibitz, and S. Avidan, "Diving into haze-lines: Color restoration of underwater images," in *Proc. British Machine Vision Conference (BMVC)*, vol. 1, no. 2, 2017.

[381] D. Akkaynak and T. Treibitz, "Sea-thru: A method for removing water from underwater images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1682–1691.

[382] Z.-u. Rahman, D. J. Jobson, and G. A. Woodell, "Multi-scale retinex for color image enhancement," in *Proceedings of 3rd IEEE international conference on image processing*, vol. 3, 1996, pp. 1003–1006.

[383] K. Iqbal, M. Odetayo, A. James, R. A. Salam, and A. Z. H. Talib, "Enhancing the low quality images using unsupervised colour correction method," in *2010 IEEE International Conference on Systems, Man and Cybernetics*, 2010, pp. 1703–1709.

[384] X. Sun, L. Liu, Q. Li, J. Dong, E. Lima, and R. Yin, "Deep pixel-to-pixel network for underwater image enhancement and restoration," *IET Image Processing*, vol. 13, no. 3, pp. 469–474, 2019.

[385] C. Li, S. Anwar, and F. Porikli, "Underwater scene prior inspired deep underwater image and video enhancement," *Pattern Recognition*, vol. 98, p. 107038, 2020.

[386] J. Li, K. A. Skinner, R. M. Eustice, and M. Johnson-Roberson, "WaterGAN: Unsupervised generative network to enable real-time color correction of monocular underwater images," *IEEE Robotics and Automation letters*, vol. 3, no. 1, pp. 387–394, 2017.

[387] C. Li, J. Guo, and C. Guo, "Emerging from water: underwater image color correction based on weakly supervised color transfer," *IEEE Signal Process. Lett.*, vol. 25, no. 3, pp. 323–327, 3 2018.

[388] Y. Guo, H. Li, and P. Zhuang, "Underwater image enhancement using a multi-scale dense generative adversarial network," *IEEE Journal of Oceanic Engineering*, vol. 45, no. 3, pp. 862–870, 2019.

[389] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*, 2016, pp. 694–711.

[390] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1501–1510.

[391] P. K. Sharma, I. Bisht, and A. Sur, "Wavelength-based Attributed Deep Neural Network for Underwater Image Restoration," *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6 2021. [Online]. Available: http://arxiv.org/abs/2106.07910

[392] C. Li, C. Guo, W. Ren, R. Cong, J. Hou, S. Kwong, and D. Tao, "An underwater image enhancement benchmark dataset and beyond," *IEEE Transactions on Image Processing*, vol. 29, pp. 4376–4389, 2019.

[393] K. Zuiderveld, ""Contrast Limited Adaptive Histogram Equalization"," in *"Graphic Gems IV"*. San Diego: "Academic Press Professional", 1994, pp. 474–485.

[394] Y.-T. Peng and P. C. Cosman, "Underwater Image Restoration Based on Image Blurriness and Light Absorption," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1579–1594, 4 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7840002/

[395] D. Huang, Y. Wang, W. Song, J. Sequeira, and S. Mavromatis, "Shallow-water image enhancement using relative global histogram stretching based on adaptive parameter acquisition," in *International conference on multimedia modeling*, 2018, pp. 453–465.

[396] H. Li, J. Qin, X. Xiang, L. Pan, W. Ma, and N. N. Xiong, "An Efficient Image Matching Algorithm Based on Adaptive Threshold and RANSAC," *IEEE Access*, vol. 6, pp. 66 963–66 971, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8528451/

[397] P. A. Castrillo, C. Varela-Dopico, R. Bermúdez, P. Ondina, and M. I. Quiroga, "Morphopathology and gill recovery of Atlantic salmon during the parasitic detachment of Margaritifera margaritifera," *Journal of Fish Diseases*, 2021.

[398] R. I. Figueroa, A. De Bustos, and . Cuadrado, "A novel FISH technique for labeling the chromosomes of dinoflagellates in suspension," *PLoS ONE*, 2018.

[399] A. K. Powers, C. A. Garita-Alvarado, R. Rodiles-Hernández, D. J. Berning, J. B. Gross, and C. P. Ornelas-García, "A geographical cline in craniofacial morphology across populations of Mesoamerican lake-dwelling fishes," *Journal of Experimental Zoology Part A: Ecological and Integrative Physiology*, 2020.

[400] G. Sanchez-Torres, A. Ceballos-Arroyo, and S. Robles-Serrano, "Automatic Measurement of Fish Weight and Size by Processing Underwater Hatchery Images," *Engineering Letters*, vol. 26, no. 4, 2018. [Online]. Available: https://bit.ly/2Z8jw0h

[401] J. R. Mathiassen, E. Misimi, B. Toldnes, M. Bondø, and S. O. Østvik, "High-speed weight estimation of whole herring (Clupea harengus) using 3D machine vision," *Journal of food science*, vol. 76, no. 6, pp. E458–E464, 2011.

[402] R. Islamadina, N. Pramita, F. Arnia, and K. Munadi, "Estimating fish weight based on visual captured," in *2018 International Conference on Information and Communications Technology, ICOIACT 2018*, 2018.

[403] A. F. A. Fernandes, E. M. Turra, . R. de Alvarenga, T. L. Passafaro, F. B. Lopes, G. F. O. Alves, V. Singh, G. J. M. Rosa, E. R. de Alvarenga, T. L. Passafaro, F. B. Lopes, G. F. O. Alves, V. Singh, and G. J. M. Rosa, "Deep Learning image segmentation for extraction of fish body measurements and prediction of body weight and carcass traits in Nile tilapia," *Computers and Electronics in Agriculture*, vol. 170, 2020.

[404] F. Suo, K. Huang, G. Ling, Y. Li, and J. Xiang, "Fish Keypoints Detection for Ecology Monitoring Based on Underwater Visual Intelligence," in *16th IEEE International Conference on Control, Automation, Robotics and Vision, ICARCV 2020*, 2020.

[405] C. H. Tseng, C. L. Hsieh, and Y. F. Kuo, "Automatic measurement of the body length of harvested fish using convolutional neural networks," *Biosystems Engineering*, 2020.

[406] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Nature, 2016, vol. 9912 LNCS, pp. 483–499. [Online]. Available: http://link.springer.com/10.1007/978-3-319-46484-8_29

[407] L. Li, B. Dong, E. Rigall, T. Zhou, J. Dong, and G. Chen, "Marine Animal Segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 4, pp. 2303–2314, 4 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9471801/

[408] I. Bello, B. Zoph, Q. Le, A. Vaswani, and J. Shlens, "Attention augmented convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, 2019, pp. 3285–3294.

[409] H. Touvron, P. Bojanowski, M. Caron, M. Cord, A. El-Nouby, E. Grave, G. Izacard, A. Joulin, G. Synnaeve, J. Verbeek, and H. Jégou, "ResMLP: Feedforward networks for image classification with data-efficient training," *IEEE*, 5 2021. [Online]. Available: http://arxiv.org/abs/2105.03404

[410] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[411] P. Ramachandran, I. Bello, N. Parmar, A. Levskaya, A. Vaswani, and J. Shlens, "Stand-alone self-attention in vision models," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[412] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy, "MLP-Mixer: An all-MLP Architecture for Vision," *Proceedings - 2021 International Conference on Computer Vision Workshop, ICCVW 2021*, 5 2021. [Online]. Available: http://arxiv.org/abs/2105.01601

[413] M. Sandler, J. Baccash, A. Zhmoginov, and A. Howard, "Non-discriminative data or weak model? on the relative importance of data and model resolution," in *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, 2019, pp. 1036–1044.

[414] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.

[415] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation," *CoRR*, vol. abs/1801.0, 2018.

[416] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and ¡0.5MB model size," *Computer Vision and Pattern Recognition*, 2 2016. [Online]. Available: http://arxiv.org/abs/1602.07360

[417] M. Jahanbakht, W. Xiang, N. J. Waltham, and M. R. Azghadi, "Distributed Deep Learning in the Cloud and Energy-efficient Real-time Image Processing at the Edge for Fish Segmentation in Underwater Videos," *IEEE Access*, pp. 1–1, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9869810/

[418] X. Li, Y. Grandvalet, and F. Davoine, "Explicit inductive bias for transfer learning with convolutional networks," in *35th International Conference on Machine Learning, ICML 2018*, vol. 6, 2018, pp. 4408–4419.

[419] N. Cohen and A. Shashua, "Inductive bias of deep convolutional networks through pooling geometry," in *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.

[420] S. Lee and C. Lee, "Revisiting spatial dropout for regularizing convolutional neural networks," *Multimedia Tools and Applications*, vol. 79, no. 45-46, pp. 34 195–34 207, 2020.

[421] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *ICCV*, 6 2016. [Online]. Available: http://arxiv.org/abs/1606.08415

[422] B. Jiang, Z. Pan, and Y. Qiu, "Study on the key technologies of a high-speed CMOS camera," *Optik*, vol. 129, 2017.

[423] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information (Switzerland)*, vol. 11, no. 2, 2020.

[424] L. Wang, Y. Zhang, and J. Feng, "On the Euclidean distance of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1334–1339, 2005.

[425] F. Nielsen, "On a generalization of the jensen-shannon divergence and the jensen-shannon centroid," *Entropy*, vol. 22, no. 2, 2020.

[426] J. E. Contreras-Reyes and R. B. Arellano-Valle, "Kullback-Leibler divergence measure for multivariate skew-normal distributions," *Entropy*, vol. 14, no. 9, pp. 1606–1626, 2012.

[427] 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 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014.

[428] S. K. Tripathy, "Significance of Traditional and Advanced Morphometry to Fishery Science," *Journal of Human, Earth, and Future*, vol. 1, no. 3, pp. 153–166, 9 2020. [Online]. Available: https://hefjournal.org/index.php/HEF/article/view/33

[429] D. R. Jerry, D. B. Jones, M. Lillehammer, C. Massault, S. Loughnan, H. S. Cate, P. J. Harrison, J. M. Strugnell, K. R. Zenger, and N. A. Robinson, "Predicted strong genetic gains from the application of genomic selection to improve growth related traits in barramundi (Lates calcarifer)," *Aquaculture*, vol. 549, 2022.

[430] N. V. Sang, N. T. Luan, N. V. Hao, T. V. Nhien, N. T. Vu, and N. H. Nguyen, "Genotype by environment interaction for survival and harvest body weight between recirculating tank system and pond culture in Penaeus monodon," *Aquaculture*, vol. 525, p. 735278, 8 2020.

[431] C. E. Boyd, R. P. Davis, and A. A. McNevin, "Perspectives on the mangrove conundrum, land use, and benefits of yield intensification in farmed shrimp production: A review," 2022.

[432] P. Mitteroecker and P. Gunz, "Advances in Geometric morphometrics," *Evolutionary Biology*, vol. 36, no. 2, 2009.

[433] A. Setiawan, H. Hadiyanto, and C. E. Widodo, "Shrimp Body Weight Estimation in Aquaculture Ponds Using Morphometric Features Based on Underwater Im-

age Analysis and Machine Learning Approach," *Revue d'Intelligence Artificielle*, vol. 36, no. 6, pp. 905–912, 12 2022.

[434] T. T. E. Vo, H. Ko, J. H. Huh, and Y. Kim, "Overview of Smart Aquaculture System: Focusing on Applications of Machine Learning and Computer Vision," *Electronics 2021, Vol. 10, Page 2882*, vol. 10, no. 22, p. 2882, 11 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/22/2882/htmhttps://www.mdpi.com/2079-9292/10/22/2882

[435] J. V. Devi, S. G. Deo, and R. Khandeparkar, "Kronecker Product," in *Linear Algebra to Differential Equations*, 2021.

[436] D. Hung, N. H. Nguyen, D. A. Hurwood, and P. B. Mather, "Quantitative genetic parameters for body traits at different ages in a cultured stock of giant freshwater prawn (Macrobrachium rosenbergii) selected for fast growth," *Marine and Freshwater Research*, vol. 65, no. 3, p. 198, 2014. [Online]. Available: http://www.publish.csiro.au/?paper=MF13111

[437] N. Bravata, D. Kelly, J. Eickholt, J. Bryan, S. Miehls, and D. Zielinski, "Applications of deep convolutional neural networks to predict length, circumference, and weight from mostly dewatered images of fish," *Ecology and Evolution*, 2020.

[438] M. M. Hasan, P. C. Thomson, H. W. Raadsma, and M. S. Khatkar, "Genetic analysis of digital image derived morphometric traits of black tiger shrimp (Penaeus monodon) by incorporating G E investigations," *Frontiers in Genetics*, vol. 13, 10 2022. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/36338959/

[439] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the Carbon Emissions of Machine Learning," *IEEE*, 10 2019. [Online]. Available: /green-ai/publications/2019-11-lacoste-quantifying.html