Automated Bioimage Analysis of Fish Morphometry: Enhancing Biomedical and Aquaculture Research Through Deep Learning

By

NAVDEEP KUMAR

Montefiore Institute
Department of Computer Science and Electrical Engineering
UNIVERSITY OF LIÈGE

A dissertation submitted to the University of Liège in accordance with the requirements for award of the degree of DOCTOR IN PHILOSOPHY in the Faculty of Applied Sciences.

Supervised by:
Prof. Pierre Geurts
Dr. Raphaël Marée

LIÈGE 2025

ACKNOWLEDGMENTS

First and foremost, I bow in gratitude to the Almighty, whose endless blessings, guidance, and strength have enabled me to complete this thesis successfully. It is through His grace that I have had the perseverance and wisdom to navigate this journey.

I extend my deepest gratitude to my parents and my wife Seema, whose unwavering love, sacrifices, and constant encouragement have been my greatest source of motivation. Their belief in my potential has fueled my determination to strive for excellence.

I would also like to express my sincere appreciation to my supervisors Prof. Pierre Geurts and Dr. Raphaël Marée, for their invaluable guidance, insightful advice, and continuous support throughout this research. Their expertise and constructive feedback have been instrumental in shaping this work. A special note of thanks to my project coordinator, Dr. Marc Muller, whose leadership and assistance have ensured the smooth progression of my research. His dedication and encouragement have been a pillar of strength in my academic pursuits.

I extend my warmest appreciation to my former and present colleagues and fellow researchers Romain, Ulysse, Matthia, Ba Thien for their valuable discussions, encouragement, and camaraderie during this journey. Their insights and shared experiences have been a source of inspiration. My sincere thanks also go to my dear friends Prem, Rahul, Kamal, Satwant, Bonny, Priyanka, whose unwavering support, motivation, and patience have been a constant reassurance in times of challenge.

A heartfelt and special acknowledgment to the wonderful individuals from our project team Ratish, Claudia, Clara, Paulo, Alessio, Shiva, Zach, Sunil, Lucia, Leticia, whose contributions and support have played a crucial role in the successful execution of my research. Their valuable inputs, expertise, and dedication have been truly invaluable.

I would also like to extend my deepest gratitude to Prof. Gilles Louppe for his constant help and guidance in handling the GPU clusters of the University. His support has been invaluable in overcoming technical challenges and ensuring the smooth execution of computational tasks in my research. I am equally grateful to the support staff, whose cooperation and help have significantly contributed to the completion of this thesis.

As I conclude this journey, I am reminded that no achievement is a solitary endeavor. Every step I have taken has been supported by the love, guidance, and kindness of those around me. With immense gratitude, I dedicate this work to all who have helped me reach this milestone.

With a heart full of appreciation, I extend my sincerest thanks to everyone who has been a part of this journey. This accomplishment is as much yours as it is mine.

Navdeep Kumar

ABSTRACT

Biomedical research on skeletal disorders increasingly relies on small fish models like zebrafish and medaka to investigate conditions such as osteoporosis and fibrous dysplasia. These models provide insights into human skeletal pathologies and broader disorders like cancer and arthritis. Meanwhile, in aquaculture, farmed fish frequently develop skeletal deformities in the jaw, operculum, and vertebral column, compromising fish welfare, performance, and product quality. These anomalies result in significant economic losses due to manual culling. Understanding the underlying mechanisms of skeletal development in fish is crucial for improving both human health and aquaculture sustainability. This thesis explores the application of deep learning methodologies in bioimage analysis, focusing on morphometric and phenotypic studies in biomedical and aquaculture research within the framework of EU funded **BioMedAqu** project.

A thorough literature review in Chapter 3 identifies the current state-of-the-art (SOTA) image analysis methods, including conventional techniques and emerging approaches like convolutional neural networks (CNNs). Despite the advancements in these methods, significant challenges persist, including the processing of high-content and high-throughput imaging data, the limitations of traditional image analysis protocols, and the scarcity of well-annotated datasets. The thesis systematically addresses these challenges through the development and implementation of innovative deep learning models tailored for various tasks.

Chapter 4 focuses on segmenting the operculum and head regions in flourescence microscopy images of zebrafish larvae using deep learning based Convolutional Neural Networks (CNNs). This segmentation enables precise measurement of the operculum-to-head ratio, serving as a quantitative metric for assessing bone mineralization. By mitigating class imbalance with advanced loss functions and employing a two-step segmentation process, our end-to-end approach significantly enhances automated morphometric analysis.

Chapter 5 employs deep learning methods for the detection of anatomical landmarks in various data sets of fish species in both biomedical and aquaculture research. In this chapter, various regression based strategies combined with different CNN architectures are evaluated for detecting anatomical landmarks. The implemented methods provide a robust and scalable solution for bioimage analysis, enhancing landmark detection in fish species for applications in both biomedical and aquaculture fields.

Chapter 6 focuses on detecting and segmenting weak, faint, overlapping, and missing structures in 2D lateral and ventral bioimages of zebrafish larvae. Using U-Net vari-

ants with single and multi-output masks, we demonstrated that deep learning models effectively segment bone structures, particularly in lateral views. Despite challenges in ventral views, like blurred boundaries and subjective manual annotations, our model accurately identified missing structures and segmented weak, faint, and overlapping ones while showing resilience to mislabeled data. This automated approach enhances bone development studies by reducing manual effort and improving analytical consistency.

Incorporating advanced deep learning techniques, the research outlines the complexities of designing effective neural network architectures while emphasizing the importance of preserving spatial information in biomedical images. The findings indicate that existing models often struggle with class imbalance and the subjective nature of expert annotations, hindering their performance. This research is motivated by the need to create scalable, automated solutions that can facilitate bioimage analysis while improving accessibility for researchers across disciplines.

This thesis relies on a common web based open source image analysis platform to integrate the implemented methodologies, fostering collaboration and improving result reproducibility, offering valuable insights that could benefit both aquaculture practices and biomedical research.

AUTHOR'S DECLARATION

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Liège and where applicable, any partner institution responsible for the joint award of this degree.

The author acknowledges that copyright of published works contained within the thesis resides with the copyright holder(s) of those works. I give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time. The author also acknowledges that assistance from generative AI software (ChatGPT) has been utilized in accordance with fair use policies for educational purposes. Content rephrasing using ChatGPT's output has been conducted thoughtfully and with due diligence, ensuring the originality and integrity of this thesis.

I acknowledge the support I have received for my research through the provision of European Union's Marie Skłodowska-Curie Actions (MSCA-ITN GA 766347) Training Program and was later funded a scholarship ("Subside fin de thèse") from the Faculty of Applied Sciences of the University of Liège.

SIGNED: ..... DATE: .....

TABLE OF CONTENTS

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Context

In the realm of biomedical research, model fish species like zebrafish (*Danio rerio*), and medaka (*Oryzias latipes*) are highly regarded as valuable vertebrate models. They are extensively used in a variety of biomedical applications, encompassing drug testing, morphometric screening, genome editing, toxicology assessments, and behavior analysis in vertebrates [25, 27, 117, 120, 145, 156, 165, 166]. These model fish exhibit significant genetic and metabolic pathway similarities with both fish and mammals, sharing over 70% of their genes with humans [9, 75, 148, 180]. Notably, zebrafish and medaka models are particularly advantageous due to their ease of maintenance and reproduction. These model fish are raised in a controlled environment at a facility or designated laboratory, with conditions that replicate their natural habitats. Along with other technical advantages such as their small size, typically 4.5-5.5 mm at 10 days post-fertilization (dpf) [96, 107], low maintenance cost, high fecundity, and compatibility with genetic engineering tools, these fish are popular among scientists of their suitability for in vivo imaging [175, 236] (we discuss more about imaging methods in Section 3.2 of Chapter 3). The embryonic and larval stages of these animals are translucent, allowing for the application of advanced imaging technologies to observe biological processes in a living animal. This property bears great potential for biomedical research when combined with the availability of transgenic and mutant lines that allow modeling human skeletal diseases and tracking specific organs and cell types with fluorescent markers [52]. Ac-

cording to the Business Research Insights website [2] (**accessed on 2 January, 2025**), the global zebrafish model services market size was USD 434.4 million in the year 2022 and is projected to reach USD 618.23 million in 2031, with a compound annual growth rate (CAGR) of 14.4% during the forecast period.

Also, fish is recognized as a valuable source of high-quality protein and essential nutrients that are integral to a healthy human diet. Within the aquaculture industry, fish holds a primary position as the predominant source of cultivated seafood for human consumption. According to the European Commission's Ocean and Fisheries website, marine and freshwater fish constitute approximately 49% of total aquaculture production. Commonly consumed food fish species include gilthead seabream (*Sparus aurata*), meagre (*Argyrosomus regius*), and salmon (*Salmo salar*), which are saltwater species, while rainbow trout (*Oncorhynchus mykiss*) is a freshwater counterpart. In their natural habitats, such as the sea or rivers, healthy fish thrive without external interventions in terms of food and care. However, in fish farms, fish are reared within controlled or artificial environments, such as ponds, tanks, or cages, which require external care and provisioning of food. Given the escalating global demand for aquaculture products, the industry faces significant pressure to enhance its supply. To meet this demand, fish farmers adopt intensive production practices, which can result in challenges like deteriorating water quality, higher fish density per unit of water volume, and limited food availability for the fish. These factors may contribute to stressed fish, the development of physical abnormalities, and susceptibility to serious diseases [126].

To detect and classify deformities in the farmed (cultured) or the model fish, manual inspection or analysis is employed, which requires significant time and technical effort. In addition, direct physical interaction with the fish can induce fear or stress that may reflect on its behavior. Due to abnormal behavior or stress, fish can not swim or take proper diet, which can lead to poor health of the fish [126]. To improve animal welfare both in aquaculture and biomedical research, scientists are looking for methods requiring minimal manual interaction with the animals, with more focus on their health and quality of life.

Computer vision, as a non-invasive technology, is increasingly adopted by fish farmers and biomedical researchers to monitor health and behavioral changes in animals and fish with little to no physical interaction. Nowadays, computer vision based techniques employ artificial intelligence methods such as machine learning (ML) or deep learning (DL), which not only speed up the diagnosis but are also helpful in improving the accuracy of the detection. Deep learning represents a cutting-edge AI approach that empowers

computers to learn from data and perform tasks on par with human capabilities (we discuss deep learning in detail in Section 2.4 of Chapter 2). Computer vision and image processing techniques can also be helpful to speed up other routine procedures such as animal feeding [78], animal sorting, and animal counting by automatizing these tasks and with minimal physical interaction with the animals. According to the website [187] (**accessed on 12 January, 2025**), the top 10 AI and software start-up companies for the aquaculture industry have raised USD 282 million in the past 5 years, illustrating the enthusiasm for AI-based smart farming in aquaculture.

1.2 Project BioMedAqu

This thesis was carried out within the framework of the project **Aquaculture meets Biomedicine: Innovation in Skeletal Health research (1 August 2018 - 31 January 2023)** abbreviated as **BioMedAqu**. It was a Marie Sklodowska-Curie Innovative Training Network (MCSA-ITN) with the aim of creating an innovative expertise, combining research in skeletal biology of aquaculture and model fish species. Aquaculture commonly referred to as fish farming is a highly valued industry, producing quality seafood for human consumption. However farmed fish often suffer from severe skeletal deformities in their jaw, operculum and vertebral column as sketched in Figure 1.1. These skeletal anomalies usually affect fish welfare, performance and product quality. In order to secure future markets and value, without the expansion of production efforts, fish farmers and businesses are focusing on improving the morphological quality of their current production. In fact, major economic losses are directly related to the development of skeletal disorders altering the external shape of reared fish, i.e. opercular and vertebral column deformities. Fish with such deformities are rejected by the potential retailers or customers thereby representing a significant economic loss for the fish farmers [115, 205]. Such deformities require tedious technical effort and time to manually cull out deformed fish from the productive cycle; which should be done as early as possible in order to not waste resources on growing sub-optimal fish. By finding solutions to avoid the development of these deformities, fish health can be improved and economic losses to fish farmers can be reduced to substantial extent.

Meanwhile, our aging population is affected by human skeletal pathologies (such as Osteoporosis and Fibrous dysplasia) at an alarming rate which has triggered research using the tools offered by small fish models such as zebrafish (Danio rerio) or Japanese rice fish also known as medaka (Oryzias latipes). These fish species are the predominant

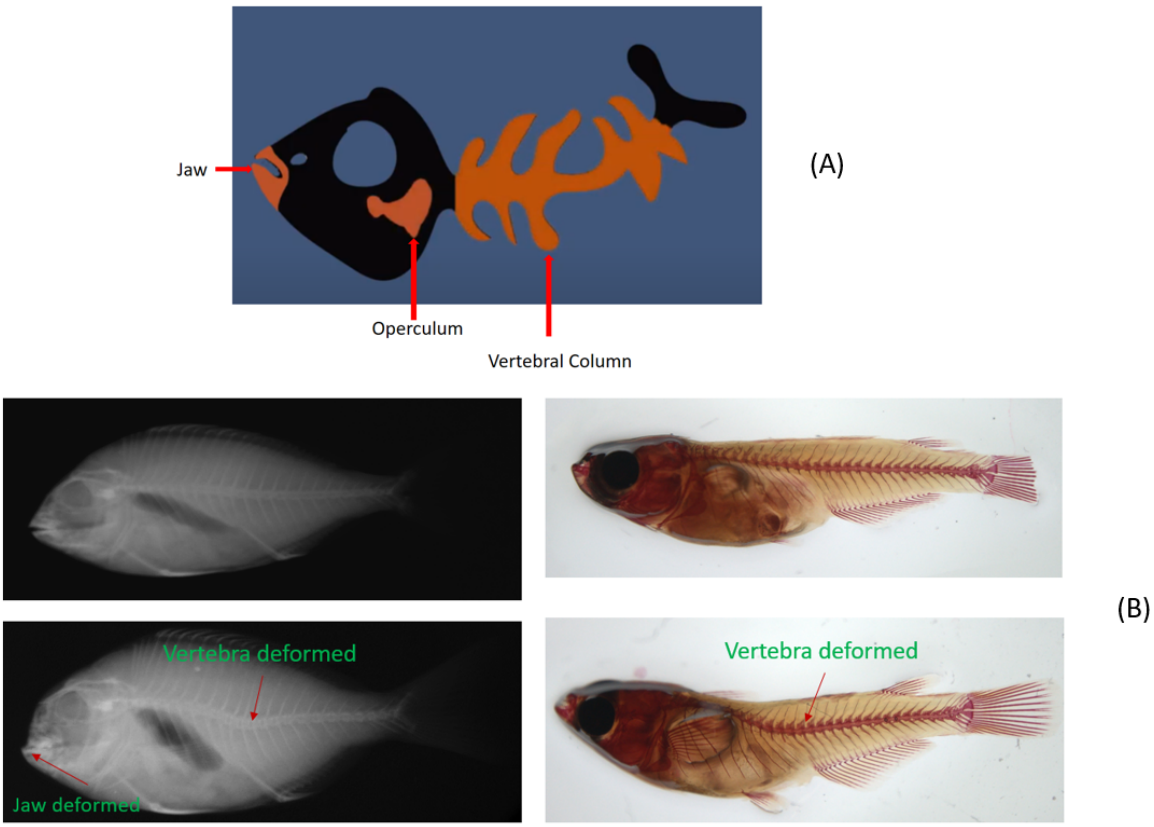

Figure 1.1: (A) Three mostly affected skeletal parts of a fish due to bone related deformities. (B) Vertebral and Jaw deformities in Gilthead Seabream (first column), vertebral deformity in Medaka (second column)

choice to be used as an animal model in the field of biomedical research to discover the potential causes of various human disorders such as development of cancer or arthritis [41, 110]. In other biomedical research areas, such as drug discovery, these model fish are used to test drugs and assess their effects on the human body [35]. Within the framework of the project, the primary focus in this biomedical research is to use these small fish models, cell culture and artificial intelligence (AI) to target the biological mechanisms that underline the development of bone disorders.

The long term goal of this project is to discover new and practical knowledge about the skeletal system of aquaculture and model fish species also enhancing the potential research for the diagnosis and treatment of bone related skeletal anomalies in human beings as well. **BiomedAqu** aims to bring together the expertise and research approaches from the aquaculture field and the biomedical sector using aquaculture and model fish species.

1.3 Thesis objective

The main objective of this thesis is to develop computer vision and AI-based deep learning algorithms tailored to automate various bioimage analysis tasks. These algorithms are designed for researchers in the biomedical and aquaculture fields, particularly for morphometric and phenotypic studies aimed at improving fish skeletal health. To accomplish this, we explore various deep learning techniques for bioimage analysis, focusing on tasks like image segmentation, anatomical landmark detection, and identification and segmentation of missing, weak, faint and overlapping bone structures. These methods are developed and evaluated with collaborators of the BioMedAqu project, which seeks to streamline the analysis of bioimages from multiple fish species such as zebrafish and medaka for biomedical research, and gilthead seabream for aquaculture applications. The algorithms are designed to support in the future fish farmers and biomedical researchers by providing faster, more accurate bioimage analysis, significantly enhancing efficiency over traditional manual methods.

1.4 Thesis structure

This thesis is organized into several chapters, the first two chapters cover the introduction and background, while the subsequent three chapters present our contributions to this work. The remainder of this manuscript is structured as outlined below:

- Chapter 2 – **Machine learning and Deep learning background**

  This chapter provides an in-depth exploration of the foundational concepts underlying machine learning and deep learning as subfields of artificial intelligence. The primary emphasis of this chapter is on deep learning methods employed in our image analysis endeavors. While not delving into exhaustive details, the objective is to provide enough background information for readers not possessing a fundamental understanding of deep learning (especially from biological background), facilitating comprehension of the contributions made in this context

- Chapter 3 - **Literature Review**

  This chapter provides a comprehensive review of current state-of-the-art computer-vision-based methods and tools, both automatic and semi-automatic, applied in image analysis for morphometric and phenotypic studies of aquaculture and biomedical model fish. It also emphasizes the primary challenges encountered in implementing these methodologies. This chapter is based on the journal paper P2, listed in Section 1.5

- Chapter 4 – **Segmentation in microscopy bioimages of zebrafish**

  This part of the dissertation describes research work done for automatic segmentation of head and operculum parts of the zebrafish larvae from single channel microscopy image dataset for the task of morphometric analysis. This chapter is based on the conference paper P4, listed in Section 1.5

- Chapter 5 - **Anatomical landmark detection in fish bioimages**

  This chapter contains the research work performed for the evaluation of various deep learning methods for the task of anatomical landmark detection on three bioimage datasets of different fish species within the framework of the BioMedAqu project. This chapter is based on the conference paper P3, listed in Section 1.5

- Chapter 6 - **Uncovering the Missing Zebrafish Larval Bone Structures: A Deep Learning Approach in Microscopy**

  This part of the thesis describes the deep learning based segmentation protocol for identifying or uncovering the missing, faint, weak and occluded structures in the lateral and ventral brighfield microscopy images of zebrafish larvae. This chapter is based on the paper P1 (in preparation for publication), mentioned in Section 1.5

- Chapter 7 - **Conclusion and future perspeftives**

  The thesis ends with this chapter, highlighting the key points from the research we carried out and discussion of some future work

1.5 Publications and Oral talks

This thesis is based on the following publications and oral talks delivered in various international conferences and Journals:

P1. Navdeep Kumar, Ratish Raman, Marc Muller, Pierre Geurts, Raphaël Marée, **“Uncovering the Bone Structures in Zebrafish Larvae: A Deep Learning Approach in Microscopy”** *(in preparation)*

P2. Navdeep Kumar, Raphaël Marée, Pierre Geurts, Marc Muller, **"Recent Advances in Bioimage Analysis Methods for Detecting Skeletal Deformities in Biomedical and Aquaculture Fish Species"**, Journal paper, Biomolecules, 2023

P3. Navdeep Kumar, Zachary Dellacqua, Claudia Di Biagio, Ratish Raman, Arianna Martini, Clara Boglione, Marc Muller, Pierre Geurts, Raphaël Marée, **"Empirical Evaluation of Deep Learning Approaches for Landmark Detection in Fish Bio-Images"**, Conference paper, European Conference on Computer Vision Workshops (ECCV- 2022)

O1. Navdeep Kumar, Marc Muller, Pierre Geurts, Raphaël Marée, **"Building Artificial Intelligence Tools for Automatic Recognition and Classification of Bone related Deformities in Aquaculture Fish"**, Oral presentation in Aquaculture Europe 2022, Rimini Italy

O2. Navdeep Kumar, Marc Muller, Pierre Geurts, Raphaël Marée, **"Deep learning based multi-modal image analysis in fish skeletal research"**. Oral presentation at Interdisciplinary Approaches in Fish Skeletal Biology, (IAFSB-2022) Olho, Algarve, Portugal.

P4. Navdeep Kumar, Alessio Carletti, Paulo J Gavaia, Leonor M Cancela, Marc Muller, Pierre Geurts, Raphaël Marée, **"Deep Learning Approaches for Head and Operculum Segmentation in Zebrafish Microscopy Images"**, Conference paper, International Conference on Computer Analysis of Images and Patterns (CAIP-2021)

O3. Navdeep Kumar, Zachary Dellacqua, Arianna Martini, Clara Boglione, Marc Muller, Pierre Geurts, Raphaël Marée, **"Towards Setting up of an Automatic Recognition System for Vertebrae and Opercular Anomalies in Reared Gilthead Seabream (Sparus aurata)"** Poster presentation in Aquaculture Europe 2021, Funchal Madeira Portugal

1.6 Code and models

The code and models are open source and are publicly available under permissive licenses:

C1. Chapter 4: https://github.com/navdeepkaushish/S_Zebrafish_Head_Opercu lum_UNet_Segmentation (code and models)

C2. Chapter 5: https://github.com/navdeepkaushish/S_Deep-Fish-Landmark-P rediction (code and models)

C3. Chapter 6: https://github.com/navdeepkaushish/S_Deep-Zebrafish-Bone-S tructures-Segmentation-V (Code)

1.7 Contributions to the datasets collected and annotated related to the thesis and the BioMedAqu project

During this research work, we have also contributed to the collection and annotation of the following data sets except D2, which is produced by GIGA research at University of Liège. Within the framework of the BioMedAqu project and according to its guidelines, these datasets are publicly available on ULiège research instance of Cytomine [124], a collaborative web-based platform developed in our team and used with collaborators from the BioMedAqu to annotate and share images. All datasets used in this thesis can be accessed using username: biomedaqu and password: BioMed$Aqu2025

D1. BIOMEDAQUE-GIGA-ZEBRAFISH-MICROSCOPIC (https://research.cytom ine.be/#/project/153858703/images) : Collected by Ratish Raman, annotated by Ratish Raman and Navdeep Kumar and used for identifying and segmenting the bone structures of zebrafish (Danio rerio) larvae in Chapter 6

D2. LANDMARKS-ULG-ZEBRA (https://research.cytomine.be/#/project/6 555554/images) : This dataset is produced by GIGA research at University of Liege and is used in Chapter 5 for anatomical landmark detection in microscopy bioimages of zebrafish (Danio rerio) larvae.

D3. BIOMEDAQUE-LESA-SAURATA-XRAYS-ZACH (https://research.cytomine.be/#/project/434321374/images) : This dataset is collected by Zachary Dellacqua, annotated by Arianna Martini and Navdeep Kumar and used in Chapter 5 for anatomical landmark detection in radiography bioimages of Gilthead seabream (Sparus aurata)

D4. BIOMEDAQU-LESA-MEDAKA (https://research.cytomine.be/#/project/549112638/images) : This dataset is collected by Claudia Di Biagio, annotated by Arianna Martini and Navdeep Kumar and used for anatomical landmark prediction in microscopy bioimages of Medaka (Oryzias latipes) fish in Chapter 5

D5. BIOMEDAQU-CCMR-FLORESCENT-ZEBRA-LARVEA-ALLESSIO (https://research.cytomine.be/#/project/144022238/images) : This dataset is collected by Alessio Carletti, annotated by Alessio Carletti and Navdeep Kumar and is used in Chapter 4 for segmenting the head and operculum areas from red channel microscopy bioimages of zebrafish

D6. BIOMEDAQU-CCMR-SUNIL-SEABREAM-MICRO (https://research.cytomine.be/#/project/542666357/images) : This dataset is collected by Sunil Poudel, annotated by Sunil Poudel and Navdeep Kumar and can be used in future projects related to identifying the skeletal deformities in Gilthead seabream (Sparus aurata)

D7. BIOMEDAQU-IPMA-AREFULL-MICRO-LETICIA (https://research.cytomine.be/#/project/535271535/images) : This dataset is collected by Leticia Luján Amoraga, annotated by Leticia Luján Amoraga and Navdeep Kumar and can be used in future projects related to identifying skeletal deformities in microscopy bioimages of Meagre (Argyrosomus regius) fish

D8. BIOMEDAQU-UGENT-XRAYS-VERTEBRAE MEASURMENTS-LUCIA (https://research.cytomine.be/#/project/527989586/images) : This dataset is collected by Lucia Drábiková and annotated by Arianna Martini and Navdeep Kumar. The dataset can be used in future studies related to finding the vertebral deformities in radiography bioimages of Salmom (Salmo salar) fish

CHAPTER 2

MACHINE LEARNING AND DEEP LEARNING BACKGROUND

This chapter provides a basic overview of AI, with a focus on key concepts in machine learning (ML) and deep learning (DL) as they relate to this research. Specifically, we emphasize deep learning techniques in computer vision, particularly convolutional neural networks (CNNs), implemented in this thesis. Rather than covering the entire AI field, this chapter summarizes relevant topics and includes references for those seeking more technical details on ML and DL. Additionally, it introduces the notation used throughout the thesis.

In Section 2.1, we start by providing brief definitions of AI and its subfields ML and DL and introducing some notation and terminologies we use in this chapter. Next, Section 2.2 covers various types of learning methods and their categories. Section 2.3 is related to protocols used for model evaluation and selection. We introduce deep learning (DL) in Section 2.4. Section 2.5 is dedicated to transfer learning and domain adaptation. Finally, Section 2.6 ends the chapter with a discussion of the metrics and loss functions used throughout this thesis.

2.1 What is Artificial Intelligence?

Artificial Intelligence (AI) is the ability of the digital computers or robotic machines to perform tasks comparably to human beings. In its simplest form, it is programmed

to execute actions and take decisions with minimum human intervention. Its more sophisticated form possesses the ability to learn from extensive datasets, whether labeled or unlabeled, and to generalize its learning experiences. This enables AI algorithms to make inferences, reason about specific situations, and autonomously engage in problem-solving [159]. Although AI is a broad field that has many sub-domains, in this thesis, we focus on statistical machine learning and deep learning concepts that are related to our research work. Figure 2.1 shows the relationship between AI, ML and DL.

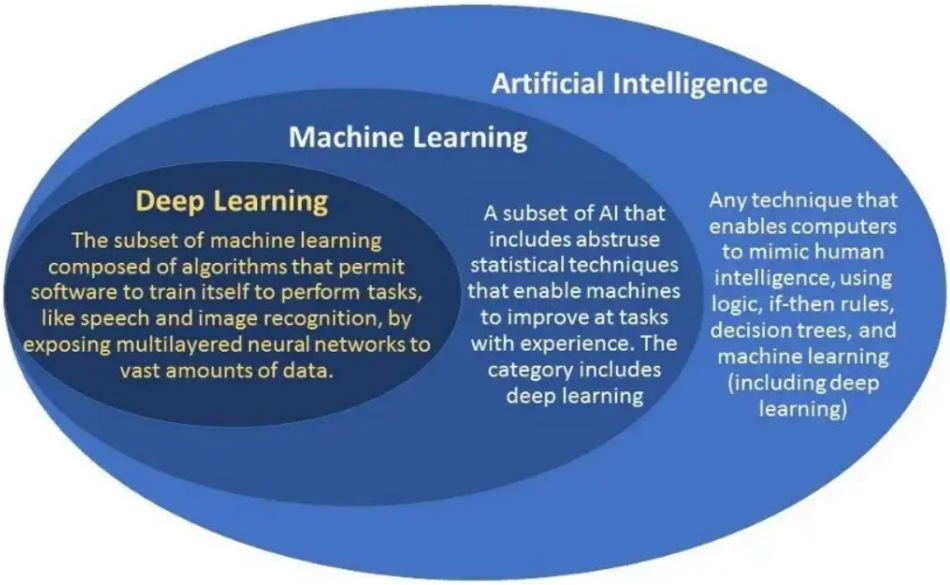

Figure 2.1: Relationship between Artificial Intelligence, Machine learning and Deep learning (source: Unite.AI).

2.1.1 Machine learning

Machine learning (ML) is a sub-field of AI in which computers are trained to optimize a performance criterion using example data or past experience. An ML model is established with specific parameters and the learning process involves executing a computer program to refine these parameters based on training data or prior experiences. The model can serve a predictive function, making future predictions, or a descriptive one, extracting knowledge from data, or even both [10].

Technically, an ML model learns from an experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E [132]. Thus Machine learning is about building programs that improve their performance on certain tasks as they gain experience,

usually in the form of exposure to data. These tasks belong to a particular class, indicating a specific domain or category of problems the program is learnt to tackle. Machine learning leverages statistical principles to construct mathematical models, primarily focusing on drawing inferences from a sample. The involvement of computer science in this domain is twofold. Firstly, during the training phase, there is a requirement for streamlined algorithms to address optimization challenges, along with the capability to handle and analyze substantial volumes of data. Secondly, after a model is acquired, the representation and algorithmic approach for inference must be optimized for efficiency. In specific scenarios, the effectiveness of the learning or inference algorithm, including its space and time complexity, can be just as crucial as its predictive accuracy [10].

2.1.2 Notations and terminologies

In the context of supervised learning (see Section 2.2.1), the output can be either **quantitative** (numerical) or **qualitative** (categorical). Categorical variables are often encoded numerically, particularly in binary classification tasks (e.g., "true" or "false," "survived" or "died"), typically using 0 and 1 or sometimes -1 and 1 [72]. These numeric representations are referred to as **targets**. For multi-class classification with K categories, a common approach is the use of **dummy variables**, where each category is represented by a K-dimensional binary vector with only one active bit.

Input variables are typically denoted by symbol X. If X is a vector, its elements can be accessed using subscripts X_j. We denote quantitative outputs as Y and qualitative outputs as C (for category). Uppercase letters such as X, Y, or C are used when referring to the general aspects of a variable. Lowercase letters are used for observed values; thus, the i^{th} observed value of X is denoted as x_i (where x_i is again a scalar or vector). Matrices are represented by bold uppercase letters; for instance, a set of N input p-vectors x_i, $i = 1,...,N$ would be expressed as the $N \times p$ matrix $\mathbf{X}$. Typically, vectors will not be represented in bold unless they consist of N components. This practice differentiates a p-vector of inputs x_i for the i^{th} observation from the n-vector $\mathbf{x}_j$, which encompasses all observations on variable X_j. As a standard assumption, all vectors are considered to be column vectors. Consequently, the i^{th} row of $\mathbf{X}$ is denoted as $\mathbf{X}_i^T$, representing the vector transpose of x_i.

Given an input vector X, the goal of supervised learning is to predict an accurate output Y, represented as $\hat{Y}$ (pronounced "Y-hat"). For real-valued outputs, $\hat{Y} \in \mathbb{R}$, while for categorical outputs, the predicted class is denoted as $\hat{C}$. Most of the classification methods however first output a prediction of the class conditional probability estimates

$\hat{Y}$ in the range $[0,1]$ that is then compared to a threshold to get a predicted class $\hat{C}$ (e.g., $\hat{c} = 1$ if $\hat{y} > 0.5$, $\hat{c} = 0$ otherwise). This methodology extends to K-level qualitative outputs, where K represents the number of categories.

2.2 Overview of learning methods

A learning algorithm in machine learning refers to the method used to train a model by adjusting its parameters, based on data. The goal of a learning algorithm is to identify patterns in the data and generalize them to make predictions or decisions on unseen data. Learning algorithms can be categorized based on the type of learning they facilitate. The process of constructing a model using a learning algorithm is termed the training phase. Subsequently, when this trained model is applied to new data, it transitions to the inference phase. In the succeeding subsections, we describe about types of learning methods used in Machine learning.

2.2.1 Supervised learning

In supervised learning (SL), the learning algorithms begins with a dataset comprising training examples, paired with their corresponding ground truth labels. For instance, in the context of learning to categorize handwritten digits, a supervised learning algorithm analyzes numerous pictures of handwritten digits, each accompanied by a label indicating the correct numerical value represented in the image. The algorithm will then learn the relationship between the images and their associated numbers, and apply that learned relationship to classify completely new images (without labels) that the machine has not seen before [122]. In another image classification task, where the objective is to assign a label to a given image, a typical example involves determining whether an image contains a cat, a dog, or another type of animal. In this context, the images are represented as integer values assigned to each image pixel, and the input space encompasses all possible color or pixel values: $\mathcal{X} \subset \mathbb{N}^{w \times h \times c}$ where w, h and c respectively denote the image width, height, and number of channels (typically 3 for RGB color images). The output space $\mathcal{Y}$ (containing all possible target values) comprises the three labels of interest: $\{cat, dog, other\}$. The model takes an image $\mathbf{X} \in \mathcal{X}$ as input and outputs $\hat{y}$, representing one of the predefined labels. It is important to note that this predicted label $\hat{y}$ might be incorrect, constituting a mistake by the model. To distinguish the correct label denoted as y (also known as ground truth) from the output label (predicted) $\hat{y}$, a

performance measure, denoted as P can evaluate the accuracy of the output label.

Formally, given a set of N training examples of the form $\{(x_1, y_1), ..., (x_N, y_N)\}$ where $x_i \in \mathcal{X}$ represents the feature vector of the i^{th} example and $y_i \in \mathcal{Y}$ is its true label (or class), a learning algorithm tries to discover a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space.

The function f is typically fit so as to minimize a "distance" metric, commonly referred to as the **loss function**, that measures how far the predictions provided by f are from the true output. In "regression" tasks, loss functions such as the **squared error** or **absolute error** are typically used to compute the error between the actual and predicted values. Conversely, "classification" tasks often rely on **log-based loss** functions, such as cross-entropy (CE) loss function. Loss functions are discussed in detail in Section 2.6.2.

In this thesis, we mainly focus on supervised learning based on DL methods, which will be introduced in Section 2.4. In the following sections, we explore some popular supervised learning approaches commonly applied to various machine learning tasks. Although these methods are not implemented in our thesis, they are included to provide readers with background information about previous techniques used for tasks similar to those we address using DL based approaches.

2.2.1.1 Linear regression

Linear regression stands as a supervised machine learning algorithm designed to determine the linear association between a dependent variable and one or multiple independent features. When there is only one independent feature, the algorithm is referred to as **Univariate Linear Regression**. Conversely, if there are more than one features involved, it is termed as **Multivariate Linear Regression**. Formally, given a vector of inputs $\mathbf{X} = (X_1, ..., X_p)$, a regression model tries to predict the output Y as:

$$\hat{Y} = \hat{\theta}_0 + \sum_{j=1}^{p} \mathbf{X}_j \hat{\theta}_j \tag{2.1}$$

where $\hat{Y}$ is called **target** variable, $\hat{\theta}_0$ is the intercept also called **bias** in machine learning. Together with $\hat{\theta}_0$, $\hat{\theta}_j$ are called the coefficients of Equation (2.1) (or **parameters**) of the regression model. A linear model approximating the relationship between the dependent and independent variables is called a regression line. This model is illustrated in Figure 2.2.

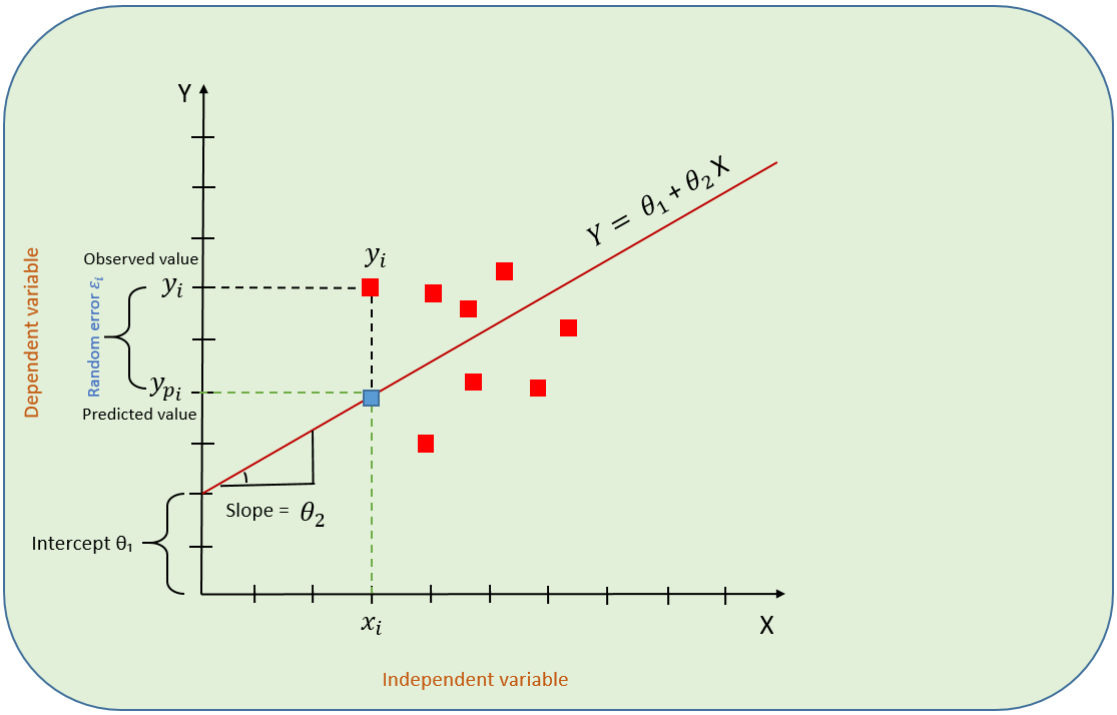

Figure 2.2: Linear regression line showing the relationship between independent variable X and dependent variable Y

2.2.1.2 Tree-based approaches

Decision tree learning [72] is a popular **non-parametric** supervised learning algorithm used for both classification and regression tasks. It represents a predictive model by a tree-like structure, similar to a flowchart, where each internal node is labeled with a test based on the input features, each branch corresponds to one of the outcomes of the test present in the branch's source node, and each leaf (terminal) node contains a value of the output, either a class in the case of classification or a numerical value in the case of regression. A prediction is computed from the tree by retrieving the value associated with the leaf reached by the test example when propagated in the tree from the root node, following the outcomes of the tests met at each internal node. An example of a classification tree is shown in Figure 2.3.

A decision tree is constructed by recursively splitting the training set into subsets until all leaf nodes contain training examples with the same value of the output (the leaf is pure) or of the inputs (no further splits are possible). At each node, the best split is found by locally maximizing some score function (based, e.g., on class entropy in the case of classification or output variance in the case of regression) measuring the quality of the

split. Fully grown trees, however, often lead to overfitting, due to the fragmentation of the data. To mitigate this, pruning is often employed to reduce tree complexity. Pruning comes in two flavors: pre-pruning, which stops splitting a node when some criterion is met (e.g., node depth is above some threshold or there are too few examples reaching the node) or post-pruning, which removes nodes from a fully grown tree so as to optimize its performance on an independent validation set. While pre-pruning is computationally more efficient, post-pruning is more effective in finding the optimal trade-off between underfitting and overfitting (see Section 2.3.2).

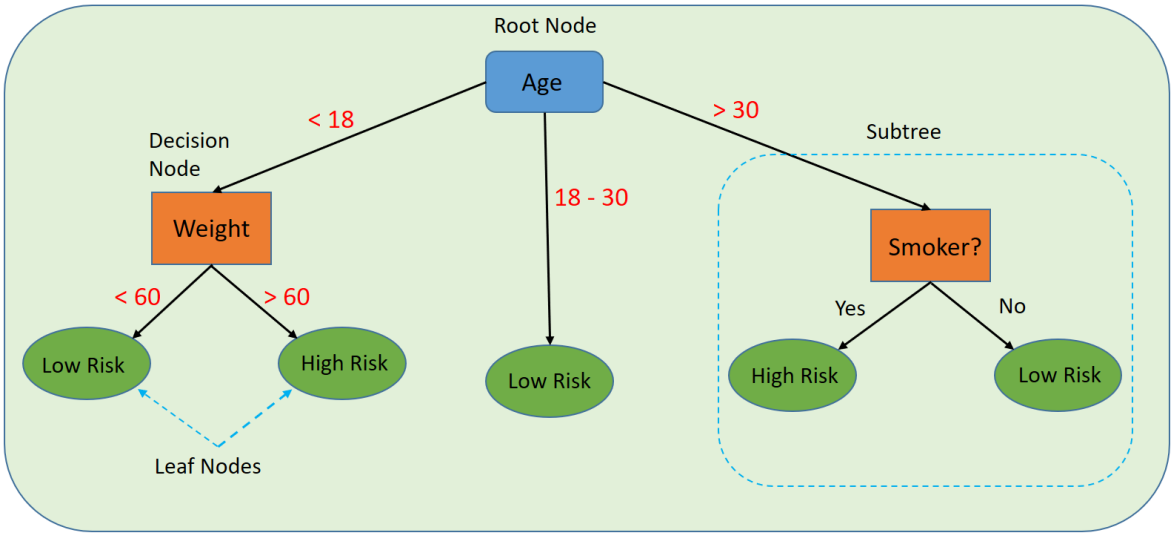

Figure 2.3: Decision tree structure for the binary classification problem of determining whether the patient is at high or low risk of a heart attack.

2.2.1.3 Ensemble methods

Ensemble methods [232] are machine learning algorithms that combine the outputs of multiple base learners in order to produce a new learner, potentially better than the individual base learners. The majority of ensemble methods employ a single base learning algorithm, producing an **homogeneous ensemble** of models all of the same type. On the other hand, some methods, such as stacking, use different types of learning algorithms to produce **heterogeneous ensembles**. Two popular families of (homogeneous) ensemble methods are averaging methods and boosting methods. *Averaging* methods build the models within the ensemble independently of each other by introducing randomization in the training procedure. The prediction of the individual models are then combined by

a simple arithmetic average in regression or majority vote in classification. The main effect of averaging methods is to reduce variance. *Boosting* methods on the other hand build the models within the ensemble sequentially, with each model focusing on the errors done by the previous models in the sequence. They are capable of turning weak, highly biased, learners (ie., marginally better than random guess in classification) into strong learners by iteratively reducing their bias.

Among popular averaging methods, one notable algorithm is *random forests*, which constructs an ensemble of decision trees. The random forests algorithm trains each tree of the ensemble from a bootstrap sample of the original training set. Each tree is furthermore learned using a modified decision tree learning algorithm that selects the best split from a subset of only k features drawn at random at each node, a technique known as feature subsampling [155]. This random selection of features at each split reduces the correlation between individual decision trees, thus enhancing the overall ensemble performance by mitigating the risk of overfitting [37, 90]. Additionally, random forests operate efficiently on large datasets and handle high-dimensional feature spaces well, making them robust to noisy or missing data [22]. The *extremely randomized trees (ET)* algorithm [62] represents a variation of random forests, introducing an additional randomization for decorrelation. The ET algorithm constructs an ensemble of unpruned decision or regression trees following the conventional top-down approach. Its primary distinctions from random forests are twofold: it randomly selects cut-points for node splitting, and it utilizes the entire learning sample to grow the trees instead of a bootstrap replica. Its explicit randomization of cut-points and attributes, coupled with ensemble averaging, aim to more effectively reduce variance compared to the other less randomized strategies, while the prime motivating factor to use original learning sample rather than bootstrap replica is to minimize the bias. Due to the simplicity of the node splitting procedure, the computational efficiency is also smaller compared to other ensemble-based methods that locally optimize cut-points.

Boosting methods include many variants such as *Gradient Boosting* [60], *AdaBoost* (Adaptive Boosting) [59], and *XGBoost* (Extreme Gradient Boosting) [38]. While generic, most boosting methods employs weak learners typically in the guise of decision trees of small depth, mostly for computational efficiency reason. In the case of Adaboost, the initial tree is trained on observations with uniform weights. Subsequently, each tree in the ensemble is trained on observations with increased (resp. decreased) weights when they are poorly (resp. correctly) classified by the preceding trees in the ensemble. Gradient boosting can handle any differentiable loss function. It builds an additive model

where each term is trained to approximate the inverse of the current gradient of the loss function. XGBoost is a specific implementation of gradient boosting with decision trees, which significantly enhance both speed and performance. XGBoost incorporates regularization techniques to prevent overfitting and enhance generalization. It also provides support for parallel processing and distributed computing, making it scalable to large datasets.

2.2.2 Unsupervised learning

Unsupervised learning (USL) is a machine learning approach where algorithms analyze and interpret data without predefined labels or categories. Unlike supervised learning, which relies on labeled datasets for training, USL discovers hidden structures and patterns within raw, unorganized data.

The primary objective of unsupervised learning is to explore the underlying distribution of data and group similar instances based on shared characteristics. The algorithm autonomously identifies clusters, associations, or anomalies without prior knowledge of correct outputs. This makes USL particularly useful in scenarios where manual labeling is impractical or costly.

In computer vision, unsupervised learning is widely used for tasks where labeled data is scarce, expensive, or difficult to obtain. Here are some real-world applications:

- **Clustering for Image Segmentation:** In this approach, image segmentation is performed by grouping the image pixels into separate regions based on their colour or intensity values. This approach is used in medical image segmentation, satellite imaging and object background separation tasks.

- **Dimensionality reduction and feature extraction:** Unsupervised dimensionality reduction techniques in computer vision streamline image data by reducing its dimensions (image compression), enhancing model performance and interpretability. The goal is to condense image features into a more manageable size while preserving critical information. Commonly applied during data preprocessing, this technique, for instance, is employed in autoencoders [208] to eliminate noise and redundant information from visual data, thus enhancing picture quality.

- **Synthetic image generation:** Unsupervised learning is employed in Generative Adversarial Networks (GANs) [67] to construct realistic or synthetic images from random noise for creating synthetic samples when real-world data is scarce or

expensive to collect. In context to medical imaging research, synthetic images are produced that enhance the performance of medical AI models by generating rare disease samples for training.

2.2.3 Semi-supervised learning

Semi-supervised learning (SSL) techniques become particularly valuable when acquiring a substantial quantity of labeled data proves to be challenging or costly, while obtaining significant amounts of unlabeled data remains comparatively easy. In these situations, neither fully supervised nor unsupervised learning approaches offer satisfactory solutions. Training with few labeled examples implies the algorithm has to deal with labeled datapoints differently than with unlabeled datapoints. For labeled points, the algorithm will use traditional supervision to update the model weights; and for unlabeled points, the algorithm minimizes the difference in predictions between other similar training examples. Formally, the given dataset $\mathbf{X} = (x_1, ..., x_n)$ is divided into two parts: the set of points $\mathbf{X}_l = (x_1, ..., x_l)$ (with $l < n$) represents instances for which corresponding labels $\mathbf{Y}_l = (y_1, ..., y_l)$ are provided, while the set $\mathbf{X}_u = (x_{l+1}, ..., x_{l+u})$ (with $l + u = n$) comprises points having no labels [36].

Semi-supervised learning depends on specific assumptions regarding the unlabeled data employed for training the model and the relationships between data points belonging to different classes. An essential requirement for the implementation of semi-supervised learning is that the unlabeled examples used in model training should be related to the task the model is being trained for. In more formal terms, SSL necessitates that the distribution $p(x)$ of the input data encompasses information about the posterior distribution $p(y|x)$, signifying the conditional probability of a given data point x belonging to a specific class y [234]. To illustrate this, if unlabeled data is employed to enhance the training of an image classifier distinguishing between cats and dogs, the training dataset must include images of both cats and dogs; images of unrelated objects such as horses or buses would not contribute to the learning process. Generally speaking, any semi-supervised learning algorithm relies on one or more of the following assumptions being explicitly or implicitly satisfied [233]:

- The **cluster assumption** states that data points within a specific cluster, defined as a group of data points sharing greater similarity among themselves than with other available data points, are likely to belong to the same class.

- The **continuity assumption** states that if two data points, x and x', are close to each other in the input space (the set of all possible values for x), then their labels, y and y', should be the same.
- The data are positioned approximately on a **manifold** with a significantly lower dimension than the input space. This assumption enables the use of distances and densities defined specifically on the manifold.

2.3 Model selection and evaluation

Evaluating a model is a fundamental principle in machine learning, which aims to understand how well the model generalizes to unseen data and whether it meets the requirements of the problem at hand. It is the process of assessing the performance and effectiveness of a machine learning model using various metrics and techniques. In Section 2.3.1, we discus about the basic criteria for assessing the model performance. Bias-variance trade-off for effective evaluation is described in Section 2.3.2 and Section 2.3.3 is dedicated to discussing some of the practical considerations and current practices involved while performing model evaluation.

2.3.1 Empirical risk minimization

In the realm of supervised learning, empirical risk minimization (ERM) is a principle that guides the process of model training by minimizing the **empirical risk**, which is essentially the error or discrepancy between the predictions made by a model and the actual observed outcomes in the training data [203]. Directly minimizing the **expected risk** also called generalization error is not feasible due to the unavailability of true distributions. It is more practical to devise an unbiased estimator of empirical risk or estimated risk, which is assessed using the provided supervised training set, $\mathcal{D}$. In ERM, the goal is to find the model parameters that minimize a certain objective function, often referred to as the loss function. This function quantifies how well the model performs on the training data.

$$R_{emp}(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f_\theta(x_i)) \tag{2.2}$$

where θ are model parameters, N is number of samples in the training data. The training example (x_i, y_i) comes from the training set $\mathcal{D}$, where x_i is the input and y_i is the corresponding true output label. $f_\theta(x_i)$ is the output produced by the model with

parameters θ given input x_i and L is the loss function that measures the discrepancy between the predicted output $f_\theta(x_i)$ and the actual output label y_i.

While ERM aims to minimize the empirical risk on the training data, the ultimate goal is to develop models that generalize well on unseen data. Generalization refers to the ability of a model to perform accurately on new, unseen examples beyond the training set. Ensuring good generalization involves not only minimizing the empirical risk but also controlling the model's complexity to prevent overfitting [197, 207].

2.3.2 Bias variance trade-off

Bias-variance trade-off refers to the term that addresses the issue of balancing the two sources of errors (i.e., **bias** and **variance**) while assessing the performance of ML model. Bias refers to the error introduced by approximating a real-world problem with a simplified model. A model with high bias tends to make strong assumptions about the underlying data distribution, leading to systematic errors [206]. High bias results in **underfitting**, where the model fails to capture the complexity of the underlying data distribution and performs poorly both on the training data and unseen data. Variance refers to the model's sensitivity to fluctuations in the training data. A model with high variance is overly sensitive to the training data and captures noise or random fluctuations in the data. High variance results in **overfitting**, where the model learns to fit the training data too closely, capturing noise and irrelevant details that do not generalize well to unseen data.

If a model is too simple (high bias), it may not capture important patterns in the data, leading to underfitting. However, increasing model complexity to reduce bias may lead to higher variance and overfitting. Achieving good model performance involves finding the right balance between bias and variance, resulting in optimal performance on unseen data. Techniques such as regularization, cross-validation, and model selection help manage the bias-variance trade-off by controlling model complexity and tuning model parameters. Figure 2.4 shows the classical depiction of bias-variance trade-off.

2.3.3 Model selection and evaluation practices

Evaluating a model based on the bias-variance trade-off may not be effective in practical scenarios, as the evaluation of these terms is non trivial and computationally intensive. Instead, one uses an independent test set, a subset of data reserved exclusively for evaluating a trained model, to estimate its true generalization performance. Two key

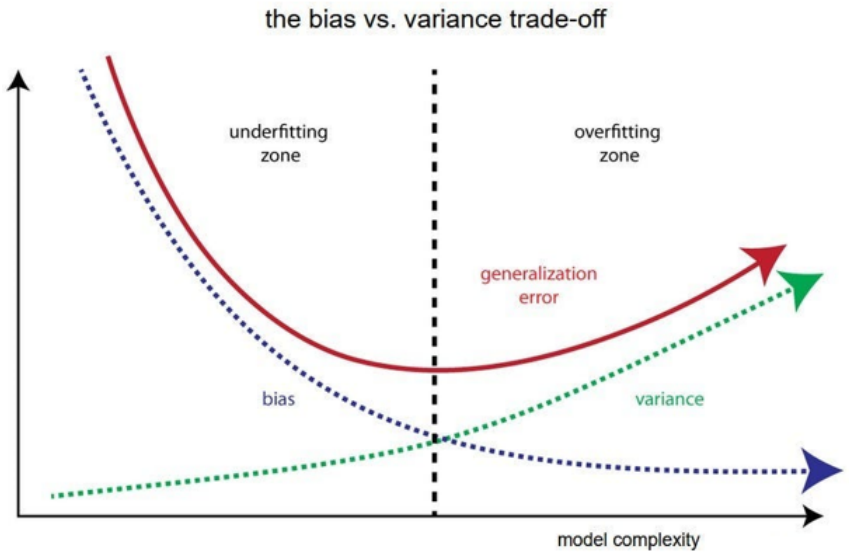

Figure 2.4: Illustration of bias-variance trade-off (source: [204])

tasks are involved in this process: **model selection** and **model evaluation**. Model selection involves choosing the best-performing model from a set of candidates based on their ability to generalize well to unseen data. Model evaluation, on the other hand, focuses on measuring the performance of the selected model using the independent test set. Various approaches can be employed to address these tasks effectively.

The first approach involves splitting the entire dataset into training and test sets in proportions, that ensure the training set is sufficiently large to train the model to an optimal level, while the test set is large enough to reliably evaluate the model. For large datasets, this may seems plausible. But in many real world problems such as in biomedical research, where the datasets are small, using this approach does not work well. To handle the small datasets, another popular approach called **cross validation** is applied for selecting and evaluating the model. In this approach, dataset is divided into multiple subsets, known as folds, where the model is trained on all folds except one and evaluated on the remaining fold. This process is repeated multiple times, with each fold serving in turn as test set in different iterations. The most common type of cross-validation is k-fold cross-validation, where the dataset is divided into k equal-sized folds. The specific way these folds are used depends on the task at hand. In a typical setting, the data is first split into k equally sized folds. In each iteration, one fold is used as the test set, while the remaining $k-1$ folds are merged and shuffled to form the training and validation sets, often split in a $(k-2):1$ ratio. The validation set is

used to select the best model, and the test set for that iteration is used to evaluate the model's performance. Once all k iterations are completed, the final score is computed by averaging the test scores from each iteration. This approach ensures that every sample in the dataset gets the opportunity to serve as a test sample, thereby reducing potential bias and providing a robust estimate of the model's generalization performance.

2.4 Deep Learning

Deep learning is is a sub-field of Artificial Intelligence in which learning is achieved from the data itself using artificial neural networks. In contrast to conventional machine learning methods where hand-crafted features are explicitly provided to the learning algorithm for model training, deep learning generates its own set of features for learning through the provided data and its labels (in case of supervised deep learning). In this section, we explore advanced deep learning concepts particularly relevant to our research work. In Section 2.4.1, we introduce the concept of 'Artificial Neural Network' (ANN) and its basic components. In Section 2.4.2, we discuss about 'Convolutional Neural Networks' (CNNs) and its fundamental working mechanisms. Parameter optimization and the learning processes of CNNs is discussed in Section 2.4.3. In Section 2.4.4, we discus about various CNN architectures used for vision tasks relevant to our thesis.

2.4.1 Neural networks and its components

An artificial neural network (ANN) is an interconnected system of artificial neurons, somehow inspired by biological neurons. Each artificial neuron can be thought of as a computational unit that receives inputs, processes them, and produces an output. The connections between neurons, known as synapses, are assigned weights that determine the strength and nature of the connection. A positive weight represents an excitatory connection, while a negative weight indicates an inhibitory connection. In an ANN, neurons are organized into layers, where each layer receives inputs from the preceding layer and passes its outputs to the next.

Mathematically, an ANN model can be defined as a parameterized approximation function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is the input space (typically $\mathbb{R}^p$) and $\mathcal{Y}$ the output space (typically R^d for some integer $d \geq 1$). A neural network function, denoted as $f(\mathbf{x};\theta)$, is constructed by composing other functions $g^{(l)}(x;\theta^{(l)})$, where $l = (1,...,L)$ represents the number L of layers. The learnable parameter set θ contains all the learnable parameters

of the neural network [222]. A typical ANN, also called **Multilayer perceptron (MLP)** consists of one input layer, one or more hidden layer(s) and one output layer as shown in Figure 2.5. The process of passing the data through the NN to infer a model prediction is called **forward propagation**. In the forward propagation, the pre-activation value, noted z_j^l, of the j^{th} neuron in layer l is computed from the previous layer as follows:

$$z_j^{(l)} = \sum_{i=1}^{n_{l-1}} w_{ji}^{(l)} a_i^{(l-1)} + b_j^{(l)} \tag{2.3}$$

where n_l is the number of neurons in layer l, $w_{ji}^{(l)}$ is the weight, connecting the i^{th} neuron of layer $l-1$ to the j^{th} neuron in layer l. $a_i^{(l-1)}$ is the (post) activation value of neuron i of layer $l-1$ and $b_j^{(l)}$ is the bias term for j^{th} neuron in layer l.

The activation of the j^{th} neuron of layer l is then obtained as follows:

$$a_j^{(l)} = \phi\left(z_j^l\right) \tag{2.4}$$

where $\phi(z)$ is a pre-defined activation function such as hyperbolic tangent, sigmoid, softmax or rectifier function. Activation functions are discussed in Section 2.4.2.4.

For the entire layer, the computations can be **vectorized** as:

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \tag{2.5}$$

$$\mathbf{a}^{(l)} = \phi\left(\mathbf{z}^{(l)}\right) \tag{2.6}$$

where $\mathbf{W}^{(l)}$ ($\in \mathbb{R}^{n_l \times n_{l-1}}$) and $\mathbf{b}^{(l)}$ ($\in \mathbb{R}^{n_l}$) are the weight matrix and the bias vector of layer l, and $\mathbf{z}^{(l)}$ and $\mathbf{a}^{(l)}$ represent respectively the pre (weighted sum) and post (outputs) activation values.

The forward propagation is initialized by taking $\mathbf{a}^{(0)}$ equal to the input vector $\mathbf{x}$. The output layer takes as input the activation values of the last hidden layer $\mathbf{z}^{(L)}$ and produces the final model prediction. It takes a similar form as a hidden layer with as many neurons as there are outputs, with parameters W^{L+1} and b^{L+1}, and an activation function that depends on the nature of the output, often the identity function in regression and a softmax activation in classification (see Section 2.4.2.4).

The trainable parameters of the networks are thus the weight matrices and bias vectors of the L hidden and the output layers:

$$\theta = \left\{\left(\mathbf{W}^{(1)}, \mathbf{b}^{(l)}\right), l = 1, \ldots, L+1\right\}.$$

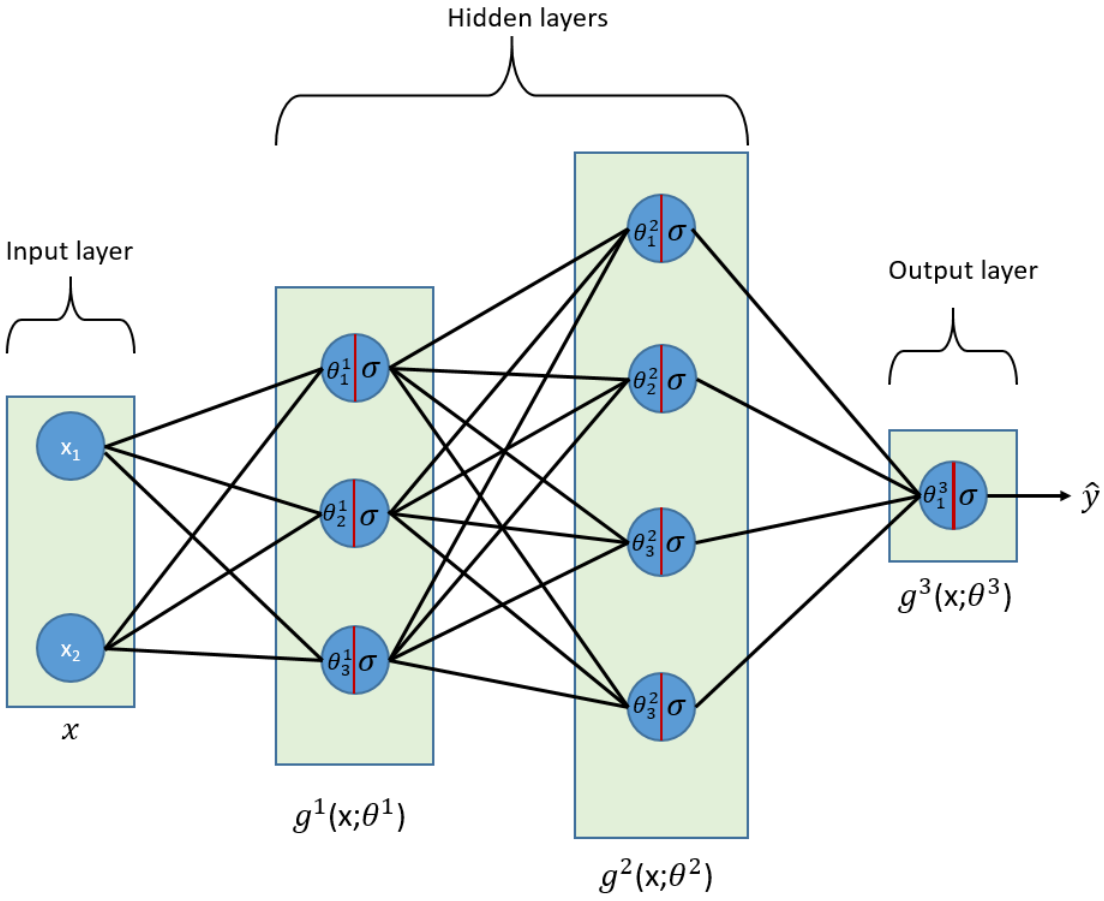

Figure 2.5: A typical ANN with one input layer x, two hidden layers (g^1,g^2) and one output layer (g^3).

2.4.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are similar to ordinary Artificial Neural Networks (ANNs) in that they are composed of neurons with weights and biases. However, CNNs include one or more convolutional layers that may be followed by pooling layers (discussed in Section 2.4.2.3), which reduce the spatial dimensions of feature maps while retaining important information. Pooling operations also contribute to making the network invariant to translations, enhancing its robustness in recognizing features regardless of their position in the input. CNNs are generally employed in vision-based tasks such as image classification, segmentation and object detection but are also used outside vision, eg., for time series analysis or language translation. CNNs share the weights across neurons that makes it possible to have less number of parameters as compared to fully connected ANNs, thus considerably reducing the complexity of the

model. Unlike ANNs, where neurons are fully connected to each other, in CNNs, the neurons are **locally connected** with a small portion of the input image. Multiple filters slide through the image to learn different features of the input image. The input to CNNs is usually a volumetric data such as images ($height \times width \times depth$) where the convolutional operation is performed across the last dimension (depth or channels). A typical CNN is composed of different types of layers and components that are described in the next section.

2.4.2.1 Convolutional layer

Convolutional layers consist of a number of filters, also called kernels, that perform convolution operations over the images and produces the **feature/activation maps**. A filter is a small matrix of weights which slides over the image from left to right and top to bottom for convolution operation. Its dimension and numbers are specified manually, thus representing a hyperparameter of the CNN. As an example, a convolution operation can be defined on an image $\mathbf{I}$ with dimension $M \times N$ with filter $\mathbf{K} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$ of dimension 2×2 as an element-wise multiplication followed by sum. For the $(i,j)^{th}$ position in the output feature map Z, the value is computed as:

$$\mathbf{z}_{[i,j]} = \sum_{p=1}^{2} \sum_{q=1}^{2} \mathbf{K}[p,q] \cdot \mathbf{I}[i+p-1, j+q-1] \tag{2.7}$$

For every pixel $\mathbf{I}[i,j]$ in the original image $\mathbf{I}$, the surrounding pixels centered around the image kernel is re-estimated and this pixel neighborhood is convolved with the kernel K resulting in a singular output value $\mathbf{z}_{[i,j]}$. The kernel can be moved across the larger image, sliding from left to right and from top to bottom, as shown in Figure 2.6.

During training, the weights of the kernels are generally randomly initialized. Current practices use some specific initialization strategies such as **"xavier initialization"** [64] or **"he initialization"** [73]. Apart from number of kernels and initialization, a kernel has other hyperparameters which are chosen or tuned manually.

- **Dimension of a filter-** A filter of size $F \times F$ applied to an image of size $I \times I \times C$ has the volume of $F \times F \times C$ (shown in Figure 2.7) that slides over the image and produces the feature or activation map of $O \times O \times 1$ (we discus about output volume O later in this section).

- **Stride-** The stride is the number of pixels by which a filter moves over the image after one operation (shown in Figure 2.8). Stride is used in 'convolution' and 'pooling' operations.

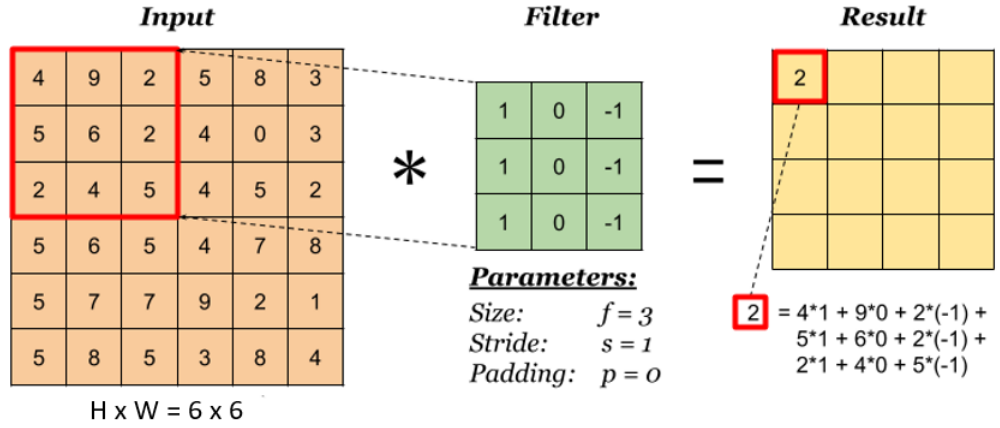

Figure 2.6: Illustration of convolution operation on an image of size 6×6 with a filter of size 3×3 [146]

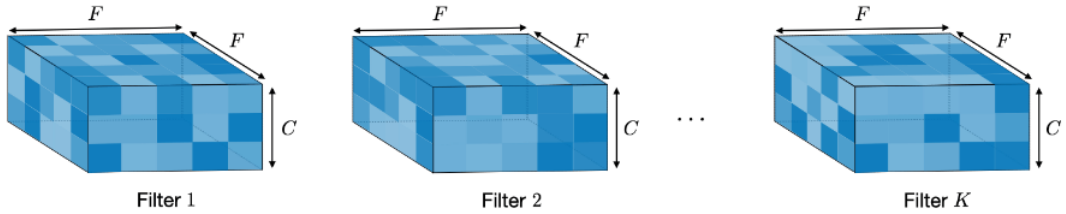

Figure 2.7: Illustration of K filters of size $F \times F \times C$ [1]

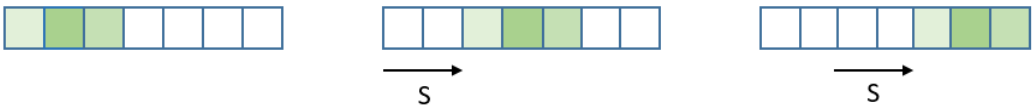

Figure 2.8: A $1d$ stride of 2, operations over 7 pixels of the image.

- **Padding-** Padding is the process of adding P zeros to the sides of the boundaries of the input image to make it customizable to the network. A **Valid** padding means no padding ($P = 0$) and the last convolution is dropped if dimensions do not match. A **Same** padding is performed in such a way that the output feature map has size $\lceil \frac{I}{S} \rceil$ where I is image dimension and S is the stride. A **Full** padding is applied where it is important for the network to apply full convolutions on the limits of the input. In the Full mode, the filter sees the inputs from end-to-end.

The output dimension of a feature map O depends upon the above mentioned hyperparameters. Considering I the length of the input volume size, F the length of the filter,

P the amount of zero padding, S the stride, then the output size O of the feature map along that dimension is given by:

$$O = \frac{I - F + P_{start} + P_{end}}{S} + 1 \tag{2.8}$$

In Equation (2.8), P_{start} and P_{end} refer to how much padding (zero-valued pixels) is added to the beginning (top/left) and end (bottom/right) of the input matrix, respectively, before applying the convolution operation in CNNs. In practice, it is usually considered $P_{start} = P_{end} = P$ in which case we can replace $P_{start} + P_{end}$ by $2P$ in Equation 2.8.

2.4.2.2 Transpose convolution

Transpose convolution also know as deconvolution is an operation in the convolution layer that act as a reverse of convolution operation. Unlike standard convolution where output feature maps has reduced dimensions, transpose convolution is applied to upsample the feature maps. Transpose convolution is used to increase the spatial resolution of feature maps. This is particularly useful in tasks such as image segmentation or image generation, where finer details need to be preserved or generated [116]. Mathematically, a transpose convolution with 2×2 kernel on an input image $\mathbf{I}$ of dimension $M \times N$ is computed for pixel (i, j) in the output image $\mathbf{O}$ as:

$$\mathbf{O}_{[i,j]} = \sum_{p=1}^{2} \sum_{q=1}^{2} \mathbf{K}[p,q] \cdot \mathbf{I}\left[\left\lfloor \frac{i-p}{s} \right\rfloor, \left\lfloor \frac{j-q}{s} \right\rfloor \right] \tag{2.9}$$

where p and q are the indices of the kernel $\mathbf{K}$ and s is the stride, typically 1 in case of transpose convolution.

Transpose convolution has the learnable parameters along with tunable hyperparametes such as padding, stride to customize the output feature maps. In practice, transpose convolution is followed by regular convolution to mitigate the artifacts of the upsampled output feature maps. It is mainly used in computer vision problems where the goal is to recover the original spatial resolution of the input images.

2.4.2.3 Pooling layer

Another basic component of CNN is **Pooling layer** which is often applied after convolutional layer and has no learnable parameters. Pooling layer is crucial to preserve spatial information and at the same time also responsible for reducing the dimension of the feature map, resulting in a compressed representation of the input feature maps. Pooling layer divides the input data into small non overlapping regions called pooling windows

and applies aggregation operation such as **Max** or **Average** of the values for each pooling window, thus reducing the size of the input feature maps. During pooling operation only height and width of the input data is reduced while depth remains unchanged. Figure 2.9 shows the 'Max' operation of the pooling layer.

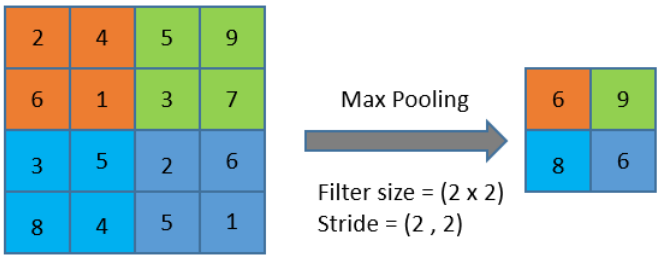

Figure 2.9: Illustration of Max pooling operation of pooling layer with filter size 2×2 and stride $(2, 2)$

Pooling Layers provide a form of **translation invariance** by extracting the most relevant features from different spatial locations, making the model more robust to variations in the position of the features. Model complexity is significantly reduced by using pooling layers as they aid in reducing the number of parameters of the network. Moreover, a form of regularization is achieved by pooling layers as they aggregate the important information from the local regions while ignoring the minor variations.

2.4.2.4 Activation layer

The activation layer employs a non-linear activation function on or before the pooling layer's output. This mechanism introduces non-linearity into the model, facilitating the learning of more complex representations of the input data. Activation function is generally applied right after the convolution operation and before pooling layer. Following are popular activation function used in CNNs:

- **ReLu-** Rectified linear unit known as 'ReLu' is a non linear activation which quashed out negative values from the input feature maps. ReLu and its variants are shown in Figure 2.10:
- **Sigmoid-** Sigmoid is a non linear activation function applied after convolution operation to squash the incoming values between $0, 1$. The output probability values are summed to 1 making it a frequent choice at the output layer of the neural network for binary classification problems. In modern days, sigmoid is not

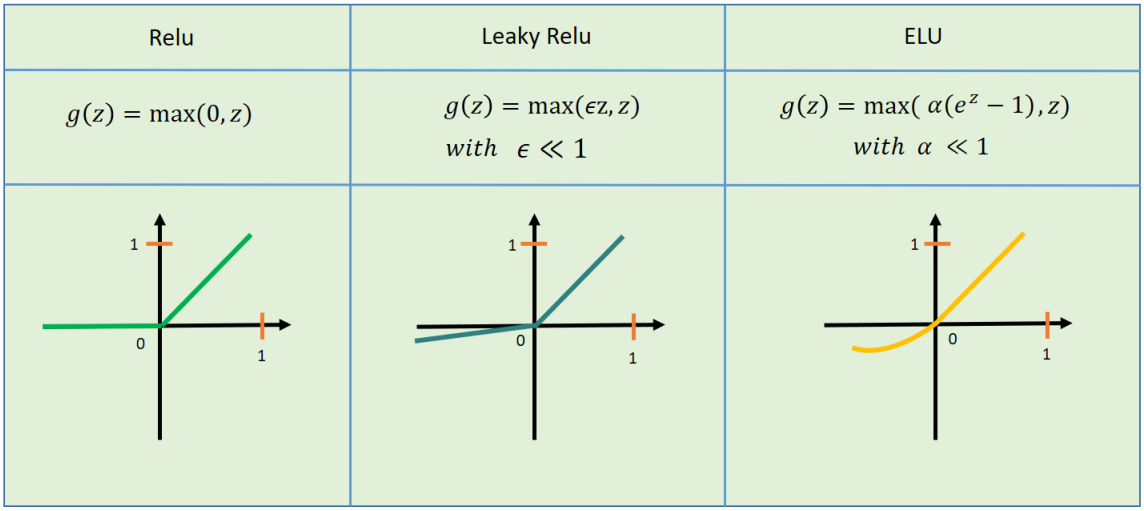

Figure 2.10: Description of rectified linear unit **'ReLu'** and its variants

used as an activation function in the middle layers of the neural network due to its propensity to produce small gradient even for large input values, thus slowing down the training with **vanishing gradient problem** [76] in large and deep networks. Mathematically, sigmoid function is expressed as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.10}$$

- **Softmax-** Softmax is a non-linear activation function which squashes the values of incoming vectored input data between $0, 1$ and so that they sum to 1. It is generally applied at the last layer of the network to output probability values between 0 and 1 for multi-class classification problems. Mathematically, if the vector x of size c is the input of the Softmax layer, its output, also of size c is computed as

$$p = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \text{ where } p_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}. \tag{2.11}$$

2.4.2.5 Fully connected layer

A fully connected layer, also known as a dense layer, is a layer that connects each neuron to all neurons in the previous layer, thus forming a subnetwork of dense connections in the CNN. It is typically applied at the end of the network and is designed to capture the global patterns of the input data. In the context of a CNN, the input to a fully connected

layer is typically the output of the preceding convolutional and pooling layers, flattened into a one-dimensional vector. Since every connection has a weight associated with it, a fully connected layers has maximum number of parameters which are learnt during training. Fully connected layers are responsible for learning high-level features and patterns in the input data, which are essential for making predictions or classifications. By connecting every feature from the previous layers to the subsequent layers, fully connected layers enable the network to learn complex relationships and make predictions based on the learned representations [17].

2.4.3 Optimization in Convolutional Neural Networks

Parameters (θ) in CNNs are randomly initialized in the beginning, meaning that they are not optimized and the performance of the network is also initially random. In order to optimize the performance of the network, these parameters need to be fine-tuned carefully so that the model gives the results as per needed. In the paradigm of supervised learning, optimization or learning is performed with a training set which contains data examples and their labels. The training data is provided to the network which has a 'loss function' that is used to minimize or maximize some objective function. Loss function is usually a mathematical function that measures the discrepancy between the predicted output of a model and the actual target value. It provides a way to quantify how well the model's predictions align with the true labels or expected outputs. For example the loss function of the form: $\mathcal{L}(f(x_i;\theta), y_i) = \|f(x_i;\theta) - y_i\|^2$ is used in linear regression and many other algorithms (we discuss about loss functions in detail in Section 2.6.2).

Since it is challenging to determine the optimal parameters analytically, **gradient descent** based numerical solutions come handy in case of optimizing CNN parameters. The goal of gradient descent is to minimize the loss function by iteratively updating the model's parameters in the direction of negative gradient of the loss function. Gradient descent start with the initial parameter set (θ_0) (weights and biases) of the network. The algorithm iteratively computes the gradient of the loss function with respect to each parameter. The gradient indicates the direction and magnitude of the steepest increase in the loss function. The parameters should thus be updated in the opposite direction of the gradient to reduce the loss. In the simplest form of gradient decent, this update is performed iteratively using the following formula:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla\mathcal{L}(\theta_t), \tag{2.12}$$

where t represent the current iteration, $\nabla\mathcal{L}$ denotes the gradient of the loss function with

respect to θ and η, known as the step size or the **learning rate**, is a hyperparameter that must be manually tuned.

During training of CNNs, choosing the appropriate learning rate is a challenging task. Setting this hyperparameter with large value makes the algorithm to diverge; setting it with very small value slows down the convergence [66]. In practice, stochastic gradient descent makes the learning rate a decreasing function η_t of the iteration number t, giving a **learning rate schedule**, so that the first iterations cause large changes in the parameters, while the later ones do only fine-tuning [174].

The gradient descent algorithm keeps on iterating training data over and over again until it reaches some convergence. Due to the high-dimensional nature of training data, it is highly unlikely that gradient descent will lead to a **global minimum** of the less function. Instead, it converges towards a **local minimum**. The standard batch mode gradient descent algorithm calculates gradients over the entire training set before updating the model parameters, which can be computationally expensive for large datasets. To address this issue, the Stochastic Gradient Descent (SGD) method updates the model parameters after computing the gradients for a single random sample from the training set in each iteration. Another computationally efficient strategy is to use **mini batch**, in which gradient descent algorithm computes the gradients of a small batch of samples from the training set before updating the model parameters. The bath size is a model hyperparameter, which has to be set manually.

Stochastic gradient descent with momentum is an improved version of regular SGD optimizer that dynamically fine-tune the model parameters during training, thus helps the algorithm converge faster while minimizing some predefined loss function. In the context of SGD optimizers, the notion of **momentum** refers to the idea to keep tracks of the direction of the past gradients with the help of **exponentially moving averages** and to use this information to update the parameters [158]. Mathematically, in SGD with momentum, the update rule for the parameters θ at iteration t is given by:

$$v_t = \beta \cdot v_{t-1} - \eta \cdot \nabla \mathcal{L}(\theta_t), \tag{2.13}$$

$$\theta_{t+1} = \theta_t + v_t, \tag{2.14}$$

where:

- v_t is moment at iteration t,
- β is a moment parameter (typically between 0 and 1),

- $\nabla\mathcal{L}(\theta_t)$ is the gradient of the loss function with respect to the parameters θ at iteration t,
- η is learning rate or step size.

In this formulation, the momentum term v_t is updated at each iteration by taking a weighted average of the previous momentum v_{t-1} and the current gradient. This weighted average acts as a memory of past gradients' directions, allowing the optimizer to continue moving in the same direction if gradients consistently point in that direction. This momentum-based update helps to smooth out the oscillations in the gradient updates. Instead of the optimizer making large, erratic jumps in the parameter space, the momentum term causes it to move more smoothly and steadily. Additionally, this smoothing effect can accelerate the convergence of the optimization process, especially in scenarios with high curvature or noisy gradients. In these challenging optimization landscapes, the momentum term can help the optimizer navigate more effectively and reach the optimum more quickly [185].

The other most widely used optimizers are **RMSProp** (Root mean squared propagation) [193] and **Adam** (adaptive moment estimation) [98] which are considered further improved versions of stochastic gradient with moment. The RMSProp optimizer works by exponentially decaying the learning rate every time the squared gradient is less than a certain threshold. This helps reduce the learning rate more quickly when the gradients become small. In this way, RMSProp is able to smoothly adjust the learning rate for each of the parameters in the network. The RMSprop algorithm utilizes exponentially weighted moving averages of squared gradients to update the parameters. Mathematically, the update rule for RMSprop at iteration t for parameter θ is given by:

$$v_t = \beta \cdot v_{t-1} + \big(1-\beta\big) \cdot \big(\nabla\mathcal{L}(\theta_t)\big)^2, \tag{2.15}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t} + \epsilon} \cdot \nabla\mathcal{L}(\theta_t), \tag{2.16}$$

where:

- v_t is the exponentially decaying average of past squared gradients for parameter θ at iteration t,
- β is a decay rate parameter (typically set to a value close to 1, e.g., 0.9),
- $\nabla\mathcal{L}(\theta_t)$ is the gradient of the loss function with respect to parameter θ at iteration t,

- η is learning rate and ϵ is s small constant for numerical stability.

Adam optimizer can be thought of as a combination of RMSProp and SGD with momentum. It uses the squared gradients to scale the learning rate like RMSprop, and it takes advantage of momentum by using the moving average of the gradient instead of the gradient itself, like SGD with momentum. This combines dynamic learning Rate and smoothing to reach the convergence. Mathematically, an adam optimizer is described as:

$$m_t = \beta_1 \cdot m_{t-1} + \left(1 - \beta_1\right) \cdot \nabla \mathcal{L}(\theta_t), \tag{2.17}$$

$$v_t = \beta_2 \cdot v_{t-1} + \left(1 - \beta_2\right) \cdot \left(\nabla \mathcal{L}(\theta_t)\right)^2, \tag{2.18}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \tag{2.19}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}, \tag{2.20}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t, \tag{2.21}$$

where:

- m_t and v_t are the first and second moment estimates of the gradients for parameter θ at iteration t,
- β_1 and β_2 are decay rate parameters (typically close to 1, e.g., 0.9 and 0.999 respectively),
- $\nabla \mathcal{L}(\theta_t)$ is the gradient of the loss function with respect to parameter θ at iteration t,
- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates of the first and second moments respectively,
- η is learning rate and ϵ is s small constant for numerical stability.

2.4.3.1 Backpropagation

All gradient descent variants discussed so far require to be able to compute the gradient of the loss function $\nabla \mathcal{L}(\theta)$ with respect to all model parameters θ, *i.e.* all weights and biases in a neural network. This is achieved in the context of neural networks using the backpropagation algorithm [158] (for backward propagation). This efficient algorithm is instrumental for the training of neural networks and its existence is one of the fundamental reasons behind their popularity.

Each individual component of the gradient, $\partial\mathcal{L}/\partial w_{ij}^{l}$, of a loss $\mathcal{L}$ measures the sensitivity of the function value (output value) with respect to a change in its argument w_{ij}^{l} (input value). In the case of MLPs or CNNs, model predictions, and consequently also the loss, are formulated as several nested function compositions, one per network layer (see Sections 2.4.1 and 2.4.2). The computation of the gradient is based on a repetitive application of the chain rule [216] that expresses the derivative of the composition of two functions f and g as follows:

$$\frac{\mathrm{d}}{\mathrm{d}x}(g(f(x))) = \frac{\mathrm{d}g}{\mathrm{d}f} \cdot \frac{\mathrm{d}f}{\mathrm{d}x}. \tag{2.22}$$

Backpropagation exploits the chain rule and the layered structure of the network to efficiently compute all components of the gradient $\nabla\mathcal{L}(\theta)$ in a single pass through the network from the last to the first layer. The resulting algorithm avoids duplicate computations and has a computational complexity linear with respect to the number of model parameters. The full mathematical derivation of the algorithm is out-of-scope of this introductory chapter but details can be found for example in [72, 147].

Backpropagation is a fundamental algorithm for training neural networks, efficiently computing gradients using the chain rule to update weights and minimize loss. Although introduced in the early 1980s, it remains the fundamental mechanism powering modern deep learning, allowing neural networks to approximate increasingly complex functions. It may however face challenges such as vanishing and exploding gradients, which can hinder training. These issues can however be mitigated through strategic choices of activation functions, proper weight initialization techniques, advanced optimization algorithms or specific network architecture. Despite its low computational complexity in theoretical term, its application to large-scale neural networks can be very costly. Fortunately, implementation on specialized hardware such as GPUs can accelerate the computation, making the training of sophisticated models practically feasible.

2.4.3.2 Batch normalization

Batch normalization [86] layer is applied in a CNN to optimize the training. It normalizes the activations of each layer by adjusting and scaling them and is typically applied after the convolution and activation layers but before pooling layers. During training, batch normalization computes the 'mean' and 'variance' for each mini-batch and uses them to normalize the activations. As an example, suppose, we have a mini-batch of activations $X = \{x^{(1)}, x^{(2)} \cdots x^{(m)}\}$ where m is the size of mini-batch, given an activation $x^{(i)}$, the first

step is to normalize the mini-batch as:

$$\hat{x}^{(i)} = \frac{x^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}, \tag{2.23}$$

where μ, σ^2 is the mean and variance of the mini-batch respectively and ϵ is a small constant for numerical stability.
After normalizing, two learnable parameters γ and β for scaling and shifting respectively are applied to the normalized activations $x^{(i)}$ as:

$$y^{(i)} = \gamma \hat{x}^{(i)} + \beta. \tag{2.24}$$

During training, the population statistics (mean and variance) are updated using exponentially moving averages as:

$$\mu_{new} = \alpha \mu_{old} + (1 - \alpha)\mu,$$
$$\sigma^2_{new} = \alpha \sigma^2_{old} + (1 - \alpha)\sigma^2,$$

where α is a momentum parameter typically close to 1 (0.9 or 0.99). During inference, it uses the aggregated statistics of the entire training dataset to normalize the activations. Overall batch normalization can speed up training by allowing the use of higher learning rates thus narrowing down the parameter search. It also helps to improve the generalization performance of the network by maintaining the intermediate activations in an acceptable range to prevent the problem of exploding or vanishing gradients [163].

2.4.3.3 Dropout layer

Dropout [176] is another useful method for optimizing the training of a CNN as it prevents overfitting by dropping some random neuron activations during training. At every iteration, it selects a random subset of neurons of the previous layer and set the activations to zero with a probability p. To ensure that the activations of different set of neurons are dropped, this process is performed for each training example and each neuron independently. The remaining neurons are scaled by a factor of $\frac{1}{1-p}$ to compensate for dropped neurons and maintains an expected output magnitude. During inference, the dropout layer is typically deactivated, with all the neuron activation being used. Overall, the dropout layer acts as a form of regularization by introducing noise into the network training. This noise helps prevent the network from relying too heavily on any particular set of neurons and encourages the network to learn more robust and generalized features.

2.4.4 CNN architectures

Currently a number of state-of-the-art CNN architectures are being used by researchers, academicians and companies to solve computer vision based image classification, segmentation and object detection tasks. We discuss some of the architectures that we have implemented in our thesis work.

2.4.4.1 Residual network

AlexNet [100] and VGG16 [171] were the first of its kinds of CNNs that have revolutionized deep learning in the field of computer vision. Nowadays they have however become obsolete and are replaced by deeper and more sophisticated CNNs. One such CNN is Residual Network, popularly known as ResNet [74], that uses skip connections or shortcuts, allowing the network to learn residual mappings instead of directly learning the desired underlying mapping. This residual mapping solves the problem of vanishing gradients that occurs in deep networks. The basic building block of ResNet is the **residual block** which consists of two convolutional layers with batch normalization and ReLu as activation functions and a shortcut or **skip connection** that skips one or more intermediate layers. This skip connection allows the flow of gradient directly through the network without passing through the activation function, thereby mitigating the vanishing gradient problem. The shortcut connections simply perform identity mapping, i.e., they directly pass along the input to the next layers. ResNet comes in number of variants, such as ResNet18, ResNet50, ResNet152 where numbers indicate the numbers of layers in the ResNet. Apart from standalone architecture, variants of ResNets are also used as backbone networks for various deep transfer learning task (We discuss about transfer learning in Section 2.5). Figure 2.11 shows the configuration of residual blocks in the ReseNet34 architecture.

2.4.4.2 Fully Convolutional Network

A Fully Convolutional Network, also known as FCN [116], is a CNN architecture primarily used for semantic segmentation tasks in computer vision. Semantic segmentation is a pixel-wise classification where each pixel is assigned a predefined label (object or background). This is in contrast with the standard image classification, where a label is assigned to the entire image. Contrary to traditional CNN, where the last layers are fully connected or dense layers, in FCN architecture, convolutional layers are implemented end-to-end fashion. In FCN, the initial layers are made up of stack of convolutional

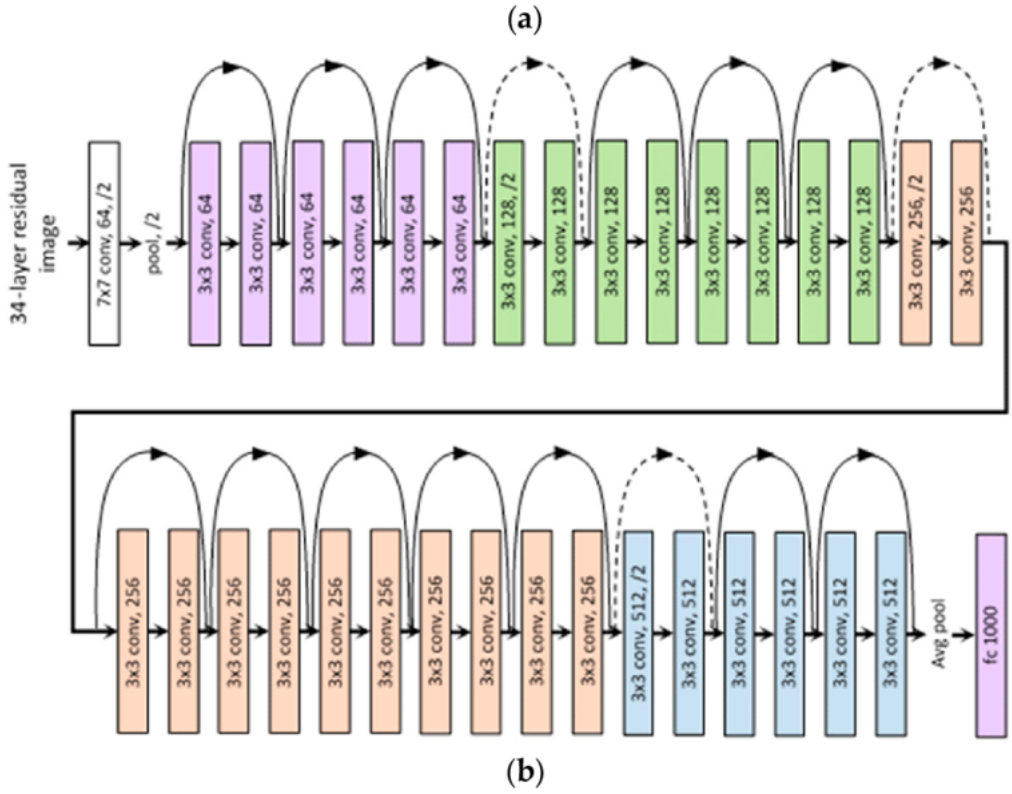

Figure 2.11: A typical 34 layered ReseNet34 architecture with 4 residual blocks. Skip connection are placed after every two convolutional layers (source:[74])

blocks followed by maxpooling whereas later layers are upsampling layers that use transpose convolutional (or decovolutional) layers to recover the spatial resolution. Skip connections are incorporated by fusion of intermediate layers with upsampling layers as shown in Figure 2.12. The variants of FCN includes FCN8, FCN16 and FCN32 where number indicates the number of layers in the network architecture.

2.4.4.3 U-Net architecture

U-Net [157] is a popular CNN architecture for semantic segmentation, initially proposed in biomedical domain. It follows an encoder-decoder architecture, where the encoder is used to extract hierarchical feature representations by gradually reducing the spatial dimensions of the input image through convolutions with max-pooling operations and the decoder part upsamples these features to generate a segmentation map with the same spatial dimensions as the input image. In the encoder part, contracting feature maps have increasing number of channels along the subsequent convolutional layers whereas in decoder part, expending feature maps have decreasing number of channels

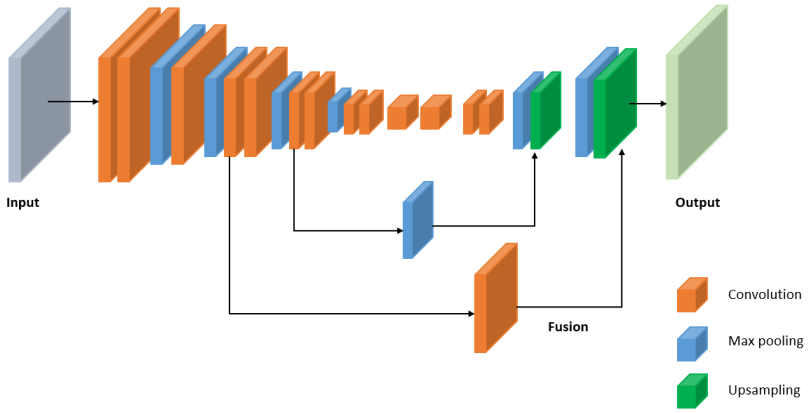

Figure 2.12: An FCN network architecture with skip (fusion) connections in the intermediate layers

along the upsampling path. Skip connections are used to fuse the compressed activations from encoder with the upsampled activations of decoder thereby combining the high-resolution, low-level features from the encoder with low-resolution, high-level features from the decoder. U-Net is proven to be very effective in semantic segmentation tasks in biomedical field, including cell segmentation, organ segmentation, and lesion detection in medical images [29, 79, 109]. Figure 2.13 shows the original U-Net architecture with all its specifications.

2.4.4.4 High resolution Network architecture

High resolution network architecture, known as HRNet [211] is primarily developed for general purpose computer vision tasks including semantic segmentation, human pose estimation, object detection and facial landmark detection. It maintains high-resolution representations throughout the network by using parallel branches with different resolutions. These branches are then fused together at multiple stages of the network, allowing the model to capture both fine-grained details and high-level semantic information simultaneously. It starts from a high-resolution convolution stream, gradually adding high-to-low resolution convolution streams one by one, and connect the multi-resolution streams in parallel. The resulting network consists of several stages as shown in Figure 2.14, and the nth stage contains n streams corresponding to n resolutions. Repeated multi-resolution fusions is performed by exchanging the information across the parallel streams over and over [211].

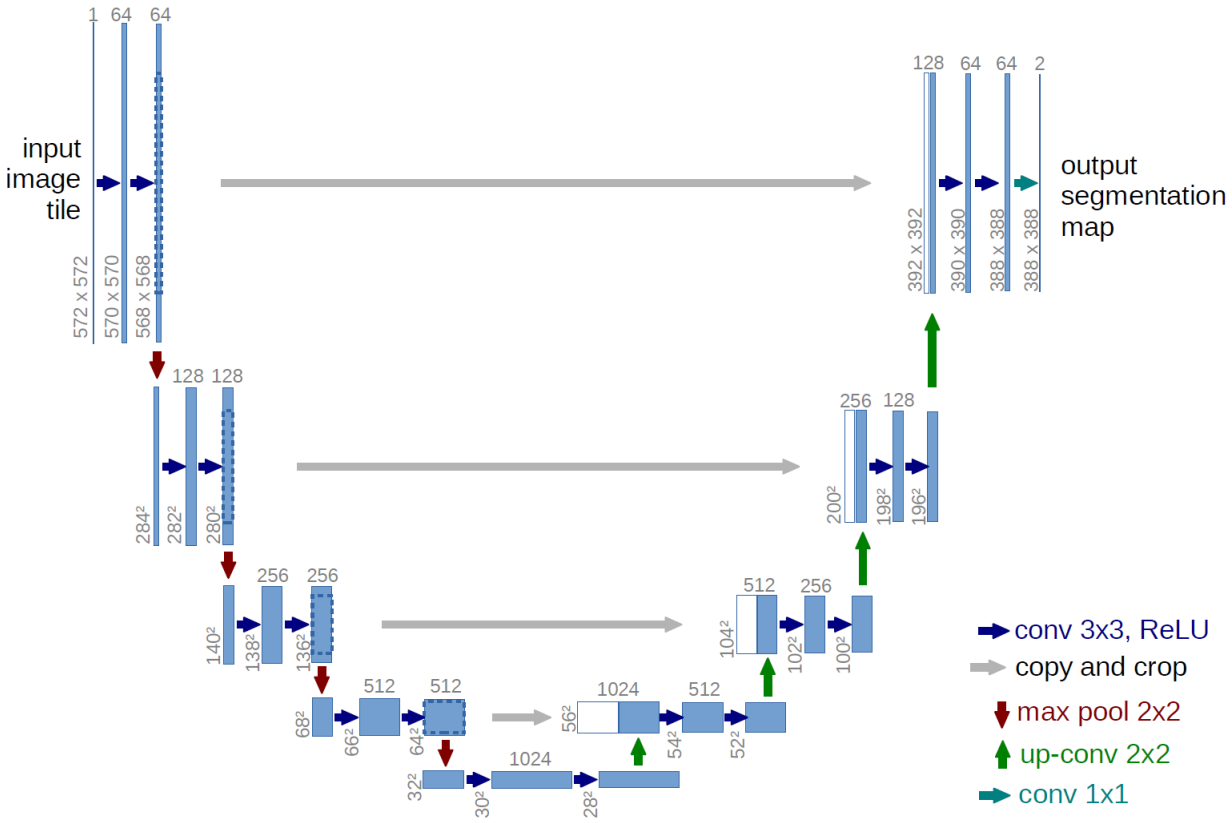

Figure 2.13: U-Net architecture with 572×572 input image resolution (source:[157])

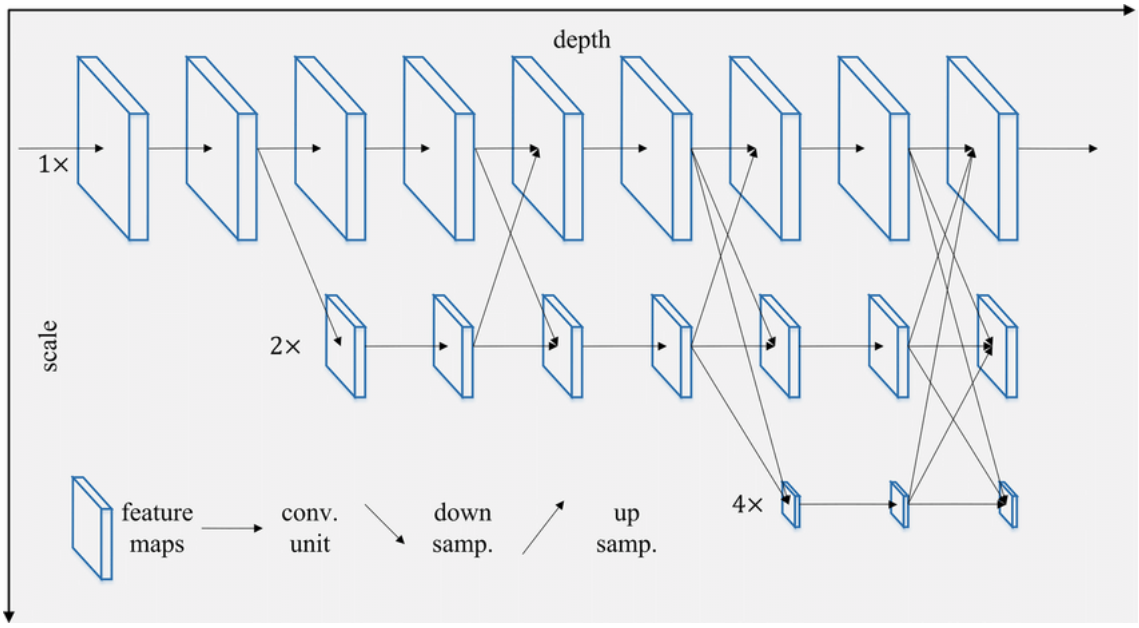

Figure 2.14: High resolution network architecture with different resolutions maintained parallelly and fused at the end (source: [211])

2.5 Deep transfer learning and domain adaptation

One of the bottlenecks in training deep CNNs is the requirement for a large amount of labeled data to train the model. In real-world applications, such as general image classification, datasets with millions of labeled images (e.g., ImageNet [49]) are available, providing sufficient data to train a CNN classifier or an object detector. In biomedical domain, where labeled image datasets are often scarce or not publicly accessible, training a CNN can be however quite challenging. Deep Transfer Learning is a machine learning technique where a pre-trained deep learning model, developed for one (source) task, is adapted for a different but related (target) task. It leverages the knowledge gained from the source task to improve performance or efficiency on the target task, particularly when the target task has limited data. By gaining the knowledge from the source task, the model can effectively learn representations that generalize well to the target task [140, 235]. In this thesis, we discuss about supervised deep transfer learning applied to anatomical landmark detection. Deep transfer learning has the following components:

- **Source and target tasks:** Source task comes from source domain $\mathcal{D}_S$ and it consists of labeled data pairs $\{(x_i^S, y_i^S)\}_{i=1}^{N_S}$ for which the model is already trained and the sample set N_S is very large. Here x_i^S are input features from source task and y_i^S are the corresponding labels. Target task comes from target domain that consists of labeled data pairs $\{(x_i^T, y_i^T)\}_{i=1}^{N_T}$ for which the model is transferred and the sample set N_T of this domain is small. In this case x_i^T are input features for target task and y_i^T are the corresponding labels (if available). In transfer learning, we leverage the model f_S learned from the source task to improve the learning process for the target task.

- **Pre-trained model:** It is a deep learning model of a source task, trained on large, diverse datasets (e.g., ImageNet for image tasks), capturing general patterns like edges, shapes, or textures in their early layers, and task-specific features in their later layers.

Deep transfer learning has two approaches to transfer the knowledge from source task to target task, namely **feature extraction** and **fine-tuning**. In feature extraction, the model f_S, learned on the source task is used as a fixed feature extractor to transform raw input data from source task into a lower-dimensional, informative feature representation. Instead of training the entire model from scratch, the early or intermediate layers (which capture general patterns) of f_S are used as a fixed, untrained, feature extractor, while a

new classifier g_T is trained using these features as inputs. More formally, let us denote by $f_S(.;\theta_S)$ the feature extractor whose parameters θ_S are determined on the source data. The target model $f_T(x;\theta_T)$ is defined as

$$f_T(x;\theta_T) = g_T(f_S(x_i^T;\theta_S);\theta_T),$$

with the parameters θ_T optimized on the target data:

$$\theta_T = \arg\min_{\theta} \sum_{i=1}^{N_T} \mathcal{L}_T\left(g_T(f_S(x;\theta_S);\theta), y_i^T\right),$$

where $\mathcal{L}_T$ is the loss function of the target task. The model g_T can be any trainable classifier (e.g., a fully connected layer).

In the fine-tuning approach, the target task model f_T is initialized with the weights of the source model f_T and then fine-tuned on the target dataset. Technically, we fine-tune the model by the following formulation:

$$f_T = \arg\min_{f} \sum_{i=1}^{N_T} \mathcal{L}_T\left(f\left(x_i^T,\theta_T\right), y_i^T\right) + \lambda \cdot \mathcal{R}(\theta_T,\theta_S), \tag{2.25}$$

where $\mathcal{L}_T$ is the loss function of the target task, $\mathcal{R}(\theta_T,\theta_S)$ is a (optional) regularization term, for instance to manage the difference between f_S and f_T. λ is the hyperparameter that controls the strength of regularization. In the fine-tuning approach, a pretrained model serves as the **backbone** or encoder of the target model f_T. The model consists of two sets of parameters: θ_S from the source task model f_S and θ_T from the target task model f_T, both of which are integrated into a common network for training. Fine-tuning offers several options for updating the model parameters [226]:

- The early layers (typically from the pretrained model) can be **frozen**, meaning their parameters remain unchanged during training, while only the later layers are optimized.
- Alternatively, the entire pretrained backbone can be frozen, and only the final classification layer is trained. The approach then becomes an instance of feature extraction.
- In a full fine-tuning approach, all parameters in θ_T are updated during optimization, allowing the model to fully adapt to the target task.

Domain adaptation is a particular family of transfer learning problems in which a model to perform a given task has been trained on one (source) data distribution

but has to be applied to perform the same task on a different, but related, (target) data distribution. Unlike in general transfer learning, in domain adaptation, the input space and tasks, and thus the label space, are shared between the source and target domains, and only the source and target input distributions are different. For instance, in biomedical research, laboratories across different regions use various data acquisition tools (such as microscopes or X-ray machines) and use all their own acquisition settings to collect similar types of data. These differences in acquisition settings often cause models trained on data acquired by one laboratory to perform poorly on data acquired by another laboratory, leading to the issue of **domain shift**. Domain shift can lead to a drop in the model's performance when applied to the target domain, as the model is trained on one distribution but tested on another. The general transfer learning strategies mentioned earlier, *i.e.* feature extraction and fine-tuning, can be applied for domain adaptation but other more specific approaches exist that try to minimize the impact of domain shift while minimizing the amount of target domain data to be collected (see, e.g., [70] for a review).

2.6 Metrics and loss functions

In the context of model evaluation, if we are interested in performance measure in terms of 'the higher the better', then it is called **metric** and if 'the lower the better', we call it **loss**. While loss functions are used during training to optimize the model, evaluation metrics (e.g., accuracy, F1-score) are used after training to measure performance on validation or test datasets. They serve different purposes but are often aligned to ensure the model is optimized for relevant metrics. They help in assessing how well our model is doing and can guide in fine-tuning the model or selecting the best model among different alternatives. In this thesis, for the segmentation tasks discussed in Chapters 4 and 6, we investigate both binary and multi-class semantic segmentation approaches which is a type of pixel-wise classification task. In these approaches, each pixel is assigned to a predefined class label, with binary segmentation involving two labels (e.g., 0 or 1), and multi-class segmentation involving more than two labels. A binary classifier typically generates a probability score $P(y = 1 \mid x)$ using **sigmoid** function in the final layer. This score ranges between 0 and 1 and represents the likelihood that the input x belongs to the positive class (e.g., "1") or background (*e.g.* , "0"). A multi-class classifier employs **softmax** function in the last layer to produce the class probabilities, ranging between 0 and 1 and summing to 1. In both cases, a **decision threshold** is a critical concept, where the goal

is to assign one of the possible labels (*i.e.* classes) to each pixel. It refers to the value at which the predicted probability or score for a given class is compared to determine the final classification. In binary classification tasks, the default decision threshold on a probability score is $T = 0.5$, meaning that inputs with a predicted probability of 0.5 or higher are classified as the positive class (*i.e.* 1) while those below this threshold are classified as the negative class (*i.e.* 0). Note however that, this probability score can be tweaked to improve confidence calibration that enable better interpretability and decision making. In case of multi-class classification, the default class is typically the one that receives the highest class probability score, but, equivalently to the use of a decision threshold, the probabilities can be rescaled using additional hyper-parameters to favor some classes over the others.

Many metrics and loss functions are available in the machine learning literature. We discuss below only those which are related to our thesis work. We fist cover in Section 2.6.1 metrics that are used to assess binary classifiers and then presents in Section 2.6.2 various loss functions that are used to train deep learning models.

2.6.1 Metrics

A metric is a quantitative measure used to evaluate the performance of a model on an independent test set. Metrics provide insights into how well a model performs with respect to the specific objectives of the problem being solved. We focus here on binary classification problems and we will denote by $y_i \in \{0,1\}$ the true output class of the ith instance and by $\hat{y}_i \in \{0,1\}$ the model prediction for the same instance. Instances in the next chapters will be mostly N image pixels that will have to be predicted as belonging to a particular class (typically the positive class, encoded as 1) or to the background (typically the negative class, encoded as 0), in the context of image segmentation tasks.

Before moving to the metrics descriptions, we first defined some terms used to calculate them.

- **True positives (TP):** the number of positive instances that are correctly predicted as positives by the model. It can be computed as:

$$TP = \sum_{i=1}^{N} \hat{y}_i \cdot y_i \tag{2.26}$$

  where N is the total number of instances (*e.g.* pixels in the image).

- **False positives (FP):** the number of negative instances that are incorrectly predicted as positives by the model. It is computed as:

$$FP = \sum_{i=1}^{N} \hat{y}_i \cdot (1 - y_i). \tag{2.27}$$

- **False negatives (FN):** the number of positive instances that are incorrectly predicted as negatives by the model, computed as:

$$FN = \sum_{i=1}^{N} (1 - \hat{y}_i) \cdot y_i. \tag{2.28}$$

- **True negatives (TN):** the number of negative instances that are correctly predicted as negative, computed as:

$$TN = \sum_{i=1}^{N} (1 - \hat{y}_i) \cdot (1 - y_i). \tag{2.29}$$

2.6.1.1 Accuracy

Accuracy is the simplest metric which tell about the model performance in term of how many predictions are correct out of total predictions. The accuracy assigns 1 to correct predictions and 0 to misclassified samples (also known as $0-1$ loss). Mathematically, the accuracy score in terms of loss can be computed by the model over a dataset of n samples as:

$$\mathcal{M}_{acc} = \frac{1}{N} \sum_{i=1}^{N} (1 - \ell_{0\text{-}1}(y_i, \hat{y}_i)) \tag{2.30}$$

where y_i and $\hat{y}_i$ are the actual and predicted values respectively. Another way of writing the accuracy formula is:

$$\mathcal{M}_{acc} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.31}$$

where TP, TN, FP and FN are defined as above. Although accuracy is the most straightforward evaluation metric, it becomes unreliable in scenarios with imbalanced datasets (see the next section). In such cases, relying on accuracy can lead to a biased model that favors the majority class, failing to generalize effectively. This is particularly problematic when the minority class holds greater significance, as the model may overlook crucial instances, leading to poor performance in real-world applications.

2.6.1.2 Precision, Recall and F1 score

Tackling class imbalance while measuring the performance of the model is vital in situations where the minority class has more weight than negative class. For example, in biomedical research, machine learning models are often used to detect whether a patient has cancer. Suppose only 2% of the patients in the dataset have cancer. In such a case, a model trained predominantly on the negative class could achieve a high accuracy of 98% simply by predicting all patients as negative, while completely ignoring the minority (positive) class. In scenarios like this, it is more meaningful to focus on metrics that account for false positives (FP) (false alarms) and false negatives (FN) (missed detections). Metrics such as **Precision**, **Recall**, and **F1 score** are more informative and reliable performance measures compared to plain accuracy. These metrics can be defined as:

- **Precision** measures the proportion of true positive predictions out of all positive predictions (both true positives and false positives). In other words, precision looks at 'how many retrieved cases are relevant'. It is also called 'positive predictive value' and calculated as:

$$Precision = \frac{TP}{TP + FP} \tag{2.32}$$

- **Recall**, also called 'sensitivity' or 'true positive rate' (TPR), measures the proportion of true positive predictions out of all actual positive instances in the dataset. In other words, it indicates the probability of retrieving relevant cases and is calculated as:

$$Recall = \frac{TP}{TP + FN} \tag{2.33}$$

- **F1 score** is the 'harmonic mean' of precision and recall values and balances both metrics. It gives more weight to lower values. The F1 score is high only if both precision and recall values are high. F1 score is calculate as:

$$F1\ Score = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} \tag{2.34}$$

In terms of TP, FP and FN, the F1 score can be expressed as:

$$F1\ Score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{2.35}$$

2.6.1.3 ROC and AUC curves

The Receiver Operating Characteristic (ROC) curve and the Area Under the ROC Curve (AUC) are commonly used evaluation metrics for binary classification models. They provide insights into a model's performance across different decision thresholds and help assess its ability to discriminate between the positive and negative classes.

- **ROC curve:** The ROC curve is a graphical representation of the performance of a binary classification model at different threshold values. The curve shows the True Positive Rate (see recall in Section 2.6.1.2), also known as sensitivity or recall, against the False Positive Rate ($FPR = \frac{FP}{FP+TN}$) or $'1 - specificity'$ at various threshold settings [186] as shown in Figure 2.15. By varying the decision threshold of the model (the probability threshold above which an instance is classified as positive), we can calculate different TPR and FPR values, resulting in points on the ROC curve. An ideal ROC curve is more inclined to top left of the plot,

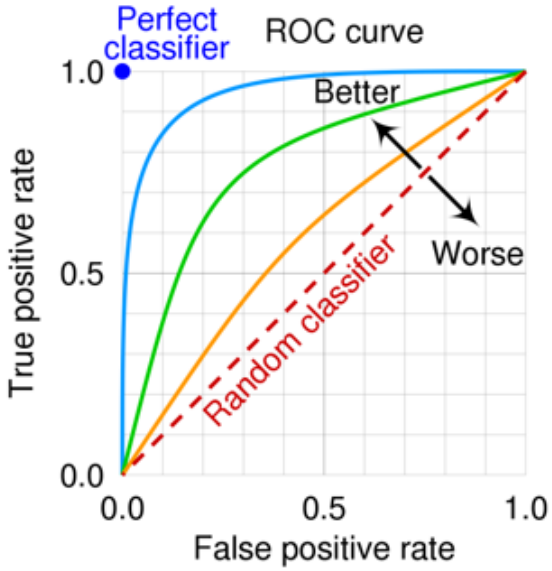

Figure 2.15: Visualization of Receiver operating characteristic (ROC) curve. (source: [237])

indicating high sensitivity and low specificity, showing better model performance. A random classifier would result in a diagonal line from the bottom-left to the top-right of the plot.

- **AUC** quantifies the overall performance of a binary classification model by calculating the area under the ROC curve. It represents the probability that the model will

rank a randomly chosen positive instance higher than a randomly chosen negative instance. AUC provides a scalar value between 0 and 1, with 1 indicating a perfect model (with $TRP = 100\%$ and $FPR = 0$) in discriminating positive and negative instances. AUC values below 0.5 shows that the model is worse than random guess and value at 0.5 indicating the model is just performing at random chances (shown in the diagonal line in Figure 2.15)

2.6.2 Loss functions

The loss function plays a crucial role in training a CNN. It is applied at the end of the last layer to quantify the difference between predicted values ($\hat{y}_i$) and the actual ground truth values (y_i). The choice of a loss function depends on the specific task at hand, such as classification, regression, or segmentation. In this section we discus some of the loss functions we have implemented in our thesis work, first for regression and then for classification problems.

2.6.2.1 Regression loss functions

Regression loss functions are used in tasks where the model output values are continuous (e.g., when predicting (x, y) positions of a landmark in an image in Chapter 5). We describe below the two regression loss functions used in our thesis.

2.6.2.1.1 Mean Squared Error (MSE). Mean squared error measures the mean squared difference between true and predicted values. It Penalizes larger errors more heavily due to the squaring and is thus sensitive to outliers. MSE is calculated as:

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.36}$$

where N is the sample size, y_i and $\hat{y}_i$ are the true and predicted values respectively.

2.6.2.1.2 Mean Absolute Error (MAE). Mean absolute error measures the average absolute difference between predicted and actual values:

$$\mathcal{L}_{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|. \tag{2.37}$$

With respect to MSE, MAE is less sensitive to outliers, *i.e.* punctual large differences between the true and predicted values.

2.6.2.2 Classification loss functions:

We review in this section the classification losses used in our thesis. Most of these losses are computed on the basis of class probability estimates instead of the final class predictions, denoted as $\hat{y}_i$ earlier. The main reason for this is that the latter are not differentiable and thus can not be optimized by gradient descent. In what follows, in the context of binary classification, we will therefore denote by $\hat{p}_i$ the probability of class 1 as predicted by the model (by default, $\hat{y}_i = 1$ if $\hat{p}_i > 0.5$, $\hat{y}_i = 0$ otherwise). In the context of multi-class classification, we will denote by $\hat{p}_{ij}$ the probability that instance i is in class j as predicted by the model and we will (one-hot) encode the true class of instance i with the variables $y_{ij} \in \{0,1\}$, such that $y_{i,j} = 1$ if instance i is in class j, $y_{ij} = 0$ otherwise.

2.6.2.2.1 Cross entropy loss. Cross entropy is one of the most popular loss functions used to train CNNs for classification tasks. It measures the distance between two discrete probability distributions: the conditional class probabilities predicted by the model, $\hat{p}_{ij}$, and the actual class probabilities. The former is the output of the network last softmax layer. The actual class probabilities are typically unknown but one uses instead a probability of 1 for the actual class of the example and 0 for all other classes, as encoded by variables y_{ij}. Let us develop cross entropy for binary classification (two classes) and multi-class classification.

- **Binary cross entropy** for a training set of N instances is computed as follows:

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \cdot \log(\hat{p}_i) + (1-y_i)\cdot \log(1-\hat{p}_i)) \tag{2.38}$$

  It is minimum, and equal to 0, when $\hat{p}_i = y_i$ for all $i = 1,\ldots,N$.

- **Categorical cross entropy** is defined as:

$$\mathcal{L}_{CCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij} \cdot \log(\hat{p}_{ij}) \tag{2.39}$$

  where C is the number of classes. As for binary cross entropy, categorical cross entropy is minimum and equal to 0 when $\hat{p}_{ij} = y_{ij}$ for all i and j.

2.6.2.2.2 Jaccard loss: The Jaccard loss function [21], also known as the Intersection over Union (IoU) loss, is commonly used in image segmentation tasks to quantify the similarity between the predicted segmentation mask and the ground truth mask. It is derived from the **Jaccard Index** (or IoU) and is designed to minimize the dissimilarity.

First, we define **Jaccard Index**, which measures the similarity between two sets. Mathematically, for two sets $\mathcal{A}$ and $\mathcal{B}$, the Jaccard Index is given by:

$$J_{(\mathcal{A},\mathcal{B})} = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|} = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A}| + |\mathcal{B}| - |\mathcal{A} \cap \mathcal{B}|}, \tag{2.40}$$

where $\mathcal{A}$ (*resp.* $\mathcal{B}$) is the set of positive pixels in the ground truth (*resp.* predicted) mask. Assuming that our N instances are the image pixels, image segmentation can be considered as a binary classification problem, where the N instances are the image pixels and the goal is to predict for each pixel whether it belongs to the segmentation mask (1) or not (0). Using our previous notations, the Jaccard Index (or IoU) can be derived as follows:

$$J(\hat{y}, y) = \frac{\sum_{i=1}^{N} \hat{y}_i \cdot y_i}{\sum_{i=1}^{N} \hat{y}_i + \sum_{i=1}^{N} y_i - \sum_{i=1}^{N} \hat{y}_i \cdot y_i}, \tag{2.41}$$

where the sum is over all N image pixels. The numerator is the number of pixels belonging to the mask ($y_i = 1$) and that are predicted as positive ($\hat{y}_i = 1$) and the denominator is the number of pixels that are either positive or predicted as positive. Because it uses the class predictions $\hat{y}_i$, the latter metrics can not be optimized by gradient descent. the Jaccard Loss used for model training is defined as follows:

$$\mathcal{L}_{Jaccard} = 1 - J(\hat{p}, y), \tag{2.42}$$

where the class predictions $\hat{y}_i \in \{0, 1\}$ are replaced in the Jaccard Index by the class probability predictions $\hat{p}_i \in [0, 1]$.

2.6.2.2.3 Dice loss: Introduced first in [181], the dice loss is used for image segmentation tasks for highly unbalanced datasets. It measures the area of overlap between predicted regions and ground truth regions, excluding the background region. To compute the dice loss, we first define the dice coefficient (using the same notations as in the previous section):

$$DC(\hat{y}, \hat{y}) = \frac{2 \cdot \sum_{i=1}^{N} \hat{y}_i \cdot y_i}{\sum_{i=1}^{N} \hat{y}_i + \sum_{i=1}^{N} y_i}. \tag{2.43}$$

The dice coefficient is very close to the Jaccard Index of Equation 2.40. Interestingly, one can show that it is also equivalent to the F1 score. Indeed, if we substitute TP, FP and FN terms from Equations 2.26, 2.27 and 2.28 respectively, into Equation 2.43, then the numerator becomes:

$$2 \cdot \sum_{i-1}^{N} \hat{y}_i \cdot y_i = 2 \cdot TP \tag{2.44}$$

and the denominator:

$$\sum_{i=1}^{N} \hat{y}_i + \sum_{i=1}^{N} y_i = TP + FP + TP + FN \tag{2.45}$$

From Equations (2.44) and (2.45), the dice coefficient can thus be rewritten as:

$$DC(\hat{y}, y) = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}, \tag{2.46}$$

which is exactly the F1 score from Equation (2.35).

As for the Jaccard index, $DC(\hat{y}, y)$ is not differentiable since it depends on class predictions. The dice coefficient in Equation 2.43 is only for model assessment. The Dice loss used for model training is written:

$$\mathcal{L}_{Dice} = 1 - DC(\hat{p}, y), \tag{2.47}$$

where again the class predictions $\hat{y}_i \in \{0, 1\}$ in the Dice Coefficient are replaced by the class probability predictions $\hat{p}_i \in [0, 1]$.

2.6.2.2.4 Tversky loss: Tversky loss [161] generalizes the dice loss by introducing two hyperparameters α and β (that should sum to 1) that allow to weight the “false positives" and “false negatives" respectively. The Tversky coefficient is calculated as:

$$Tv(\hat{y}, y) = \frac{TP}{TP + \alpha \cdot FP + \beta \cdot FN}, \tag{2.48}$$

where α and β are weighting parameters that control the relative importance of false positives (FP) and false negatives (FN), respectively. When $\alpha > \beta$, the Tversky index places more emphasis on minimizing false positives, prioritizing sensitivity (recall). When $\alpha < \beta$, tversky index tries to minimize the effect of false negatives and prioritizing precision or specificity. Dice coefficient is recovered by setting the $\alpha = \beta = 0.5$. Tversky loss is the complement of tversky index, where class predictions are substituted for class probabilities:

$$\mathcal{L}_{tversky} = 1 - Tv(\hat{p}, y). \tag{2.49}$$

Similar to the dice loss, the tversky loss quantifies the distance between two segmentation masks and is used in medical image segmentation tasks of highly imbalanced datasets (*i.e.* small segmentation masks).

2.6.2.2.5 Focal loss: Focal loss [111] is a modified version of the standard cross-entropy loss, designed to address the class imbalance problem often encountered in dense prediction tasks such as object detection, semantic segmentation, and medical

imaging. It focuses more on the hard-to-classify examples (e.g., minority classes or misclassified samples) by dynamically scaling the loss contribution of easy-to-classify (true negatives) examples. Focal loss is used as a loss function, both in binary and multi-class classification tasks. Mathematically, focal loss is defined as follows (in binary classification setting):

$$\mathcal{L}_{Focal} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \cdot \alpha \cdot (1-\hat{p}_i)^{\gamma} \cdot \log(\hat{p}_i) + (1-y_i)\cdot(1-\alpha)\cdot \hat{p}_i^{\gamma}\log(1-\hat{p}_i)\right), \tag{2.50}$$

where α and γ are two hyperparameters. With respect to binary cross entropy, α ($\in [0,1]$) is a weighting factor that allows to re-balance the classes, while γ (typically set to 2) allows to give more weights to the hard-to-classify examples. For example, when $y_i = 1$, the first term is rescaled by $(1-\hat{p}_i)^{\gamma}$, which increases as $\hat{p}_i$ moves further away from 1.

2.6.2.2.6 Bi-tempered logistic loss: Bi-Tempered Logistic Loss [11] is another extension of the standard cross-entropy loss, designed to improve robustness against label noise (mislabeling) and outliers. Unlike standard cross-entropy, which assumes data is perfectly labeled and reliable, Bi-Tempered Loss uses a tempered logarithmic and tempered exponential framework, controlled by two parameters T_1 and T_2. These parameters shape the loss to mitigate the impact of outliers and noisy labels. The Bi-Tempered Logistic Loss introduces tempered alternatives to the exponential (*e.g.* softmax function in the last layer) and logarithmic functions (*e.g.* cross entropy loss function), softening their impact and offering robustness to scenarios where the presence of outliers and mislabeling of data make the training unstable. We define **tempered logarithm function** $\log_T(x)$ as:

$$\log_T(x) = \begin{cases} \frac{x^{1-T}-1}{1-T} & \text{if } T \neq 1, \\ \log(x) & \text{if } T = 1, \end{cases} \tag{2.51}$$

When $T = 1$, it reduces to natural logarithm $\log(x)$ and if $T < 1$, the logarithm function grows slowly, thereby limiting the influence of outliers. The tempered version of exponential function $\exp_T(x)$ is defined as:

$$\exp_T(x) = \begin{cases} [1+(1-T)x]_+^{\frac{1}{1-[T]}} & \text{if } T \neq 1, \\ \exp(x) & \text{if } T = 1, \end{cases} \tag{2.52}$$

where $[z]_+ = \max(z, 0)$ ensures non-negativity. Setting $T = 1$ reduces Equation (2.52) to the standard $\exp(x)$ and for $T > 1$, it caps excessively large values, making predictions

more stable. To calculate the Bi-tempered loss, firstly, a **tempered softmax function** is used to compute the predicted probabilities $\hat{p}_{ij}$ for instance i as:

$$\hat{p}_{ij} = \frac{\exp_{T_2}(z_{ij})}{\sum_{k=1}^{C} \exp_{T_2}(z_{ik})}, \tag{2.53}$$

where z_{ij} are raw logits (model outputs) for instance i and class j and T_2 controls how the softmax function behaves. For $T_2 < 1$, the output probabilities are flattened, reducing overconfidence and for $T_2 > 1$, the probabilities are sharpened (tail-heaviness), thus keeping the loss value small while maintaining the decision boundary away from the noisy examples. The loss, for N instances, is computed using the true labels y_i and the tempered logarithm as follows:

$$\mathcal{L}_{Bi-tempered} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \cdot \log_{T_1}(\hat{p}_{ij}), \tag{2.54}$$

where T_1 controls the behavior of the logarithmic penalty. Tuning this term between 0 and 1 will ensure a finite amount of loss is incurred for each example even if they are mislabeled. It down-weights extreme probabilities, thereby mitigating the impact of noisy labels.

CHAPTER 3

LITERATURE REVIEW

In this chapter, we review various computer vision-based automatic and semi-automatic image analysis methods and tools that are used in morphometric and phenotype studies of the aquaculture and biomedical model fish. These methods and tools play a significant role in improving research by automating various aspects of the bioimage analysis. This chapter aims to provide exhaustive information about the current conventional and AI-based image analysis methods and tools to researchers from biomedical, aquaculture and computer science background. Our literature review was performed using searches in PubMed, Scopus, Google Scholar, Web of Science, Bioimage Informatics Index (https://biii.eu, and Papers With Code (https://paperswithcode.com/ databases (**accessed before 2 August 2023**), and thanks to our personal communications with researchers in the field, including members of the BioMedAqu project.

Reference: This chapter is an adapted and updated version of the work we published in "Navdeep Kumar, Raphaël Marée, Pierre Geurts, Marc Muller, **"Recent Advances in Bioimage Analysis Methods for Detecting Skeletal Deformities in Biomedical and Aquaculture Fish Species"**, Biomolecules, 2023".

3.1 Introduction

Image analysis refers to the process of examining, interpreting, and extracting meaningful information from digital images. It involves applying various algorithms, techniques, and tools to understand and analyze the content, structure, and characteristics of an

image [65]. Image analysis can encompass a wide range of tasks, including image segmentation, object detection, feature extraction, pattern recognition, image classification, and image enhancement [139]. By utilizing computer vision, machine learning, and other computational methods, image analysis enables the extraction of quantitative data, identification of patterns, and generation of insights from visual data. It finds applications in numerous fields, such as medical imaging, satellite imagery, surveillance, quality control, robotics, and scientific research, contributing to advancements in areas such as healthcare, agriculture, manufacturing [141]. By analyzing images, valuable information can be extracted, patterns can be identified, and important insights can be gained, leading to advancements in various domains. Automatic image analysis refers to the process of using AI based computer vision algorithms and techniques to analyze and extract information from images without human intervention.

In biomedical research, images also commonly referred to as *bioimages* are generated using sophisticated instruments such as x-ray machines or powerful microscopes to extract and visualize biological information within two (x,y) or three (x,y,z) dimensional coordinate spaces and four (x,y,z,t) dimensional dynamic data spaces [224]. Bioimages can provide uniquely valuable information about tissue composition, morphology and function, as well as quantitative descriptions of many fundamental biological processes. Biomedical imaging enables the real-time visualization of biological processes within living organisms, capturing alterations in receptor kinetics, molecular and cellular signaling, as well as interactions and the transit of molecules across membranes. Predominantly non-invasive, bioimaging methods provide accurate monitoring of metabolites, serving as valuable biomarkers for identifying, monitoring the progression and assessing the response to treatment of various diseases [134, 149]. While it is frequently underestimated, there exists another crucial element in imaging that could arguably be deemed the most significant dimension —the wavelength (λ) of the imaging signal. Within this pivotal dimension, specifically the electromagnetic spectrum, numerous specialized imaging technologies have emerged, utilizing various signals across the electromagnetic continuum [225]. Figure 3.1 shows commonly used bioimage modalities in biomedical research.

Proceeded by Introduction, in Section 3.2, we first highlight the popular imaging techniques used in acquiring fish bioimages. In the subsequent sections, we delve into the image analysis techniques employed in biomedical (Section 3.3) and aquaculture investigations (Section 3.4) to identify and categorize different types of bone-related deformities in both model and food fish species. Table 3.1 focuses on user-friendly, AI-

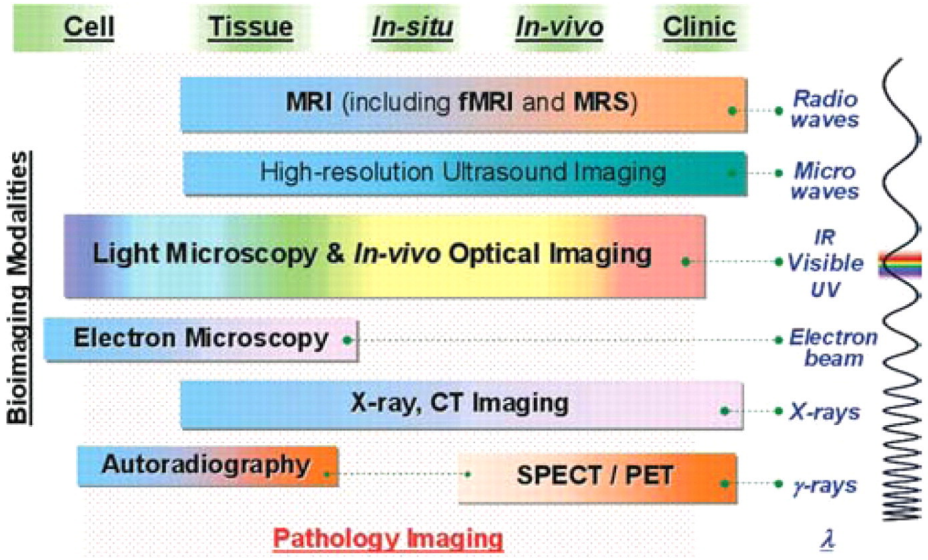

Figure 3.1: Commonly used bioimage modalities in biomedical research (source: [225])

based image analysis tools used in fish morphometric and phenotype research along with their specifications.

3.2 Imaging Techniques Used in Fish Bioimages

As mentioned in Chapter 1, one of the main advantages of using zebrafish as a model animal over other animals is its transparent body during early, external development life stages, especially from 0 to 10 days dpf. The transparent body of the model fish larva makes it easy for the biologists to see through its developing organs and bones during *in-vivo* studies and also helps to produce bioimage datasets using various image acquisition equipments [18, 82]. Given that image acquisition precedes image analysis, it is crucial to employ suitable imaging methods and protocols to ensure effective and accurate image analysis, particularly when conducting AI-based image analysis. Microscopic imaging methods necessitate a meticulous pipeline to be adhered to, ensuring the prevention of unwarranted variations in acquisition adjustments and parameters that might introduce artifacts, capable of influencing the outcomes of image analysis algorithms [130, 168]. Beyond fundamental considerations like luminosity and focus control, special attention to the fish's positioning and the characteristics of the glass plates is also needed to mitigate potential issues related to light refraction. This precautionary approach aims to prevent problems like shadowed areas in the images that could disrupt the subsequent

analysis [92]. Since most phenotype and morphometric studies in biomedical research require capturing the fine-grained information at the sub-cellular level, microscopy methods such as bright-field or fluorescence microscopy are prevalent compared to other imaging approaches [4, 101]. More recently, confocal and light-sheet microscopy deliver three-dimensional images [26], while Raman spectroscopy, Fourier-transform infrared spectroscopy, or mass spectrometry imaging are able to reveal the spatial distribution of individual (bio)molecules or classes of molecules [20, 45, 57, 77], resulting in ever more high-content and demanding analysis requirements.

Apart from microscopy methods, X-ray radiography techniques are also popular in biomedical and aquaculture research for analyzing the skeletal structures of the juvenile and adult fish, including microCT imaging [16, 50, 128]. While microscopy imaging methods are employed in the early life stages (embryonic and larval) of the model fish due to its body's optical clarity and small size, radiography methods are employed in the later life stages to visualize hard tissues. The adult model fish serves as a distinct and valuable resource for studying pathogenic and therapeutic aspects of adult human bone diseases. This is attributed to the fact that certain functions such as bone turnover, repair, degeneration, and metabolic responses are not fully mature in embryos [32]. Similarly, in aquaculture research, radiography imaging methods are utilized for juvenile and adult fish for several types of phenotype and morphometric studies [19, 48].

3.3 Fish bioimage Analysis in Biomedical Research

As outlined in Chapter 1, zebrafish or medaka is used as an animal model for many biomedical research studies such as morphometry, phenotype classifications, toxicology and drug discovery or to determine the causes of certain disease infections and pathogen dissemination [179]. Such studies involve systematic procedure and protocols such as rearing model fish in the laboratory with utmost care and supervision, preparing the fish for image acquisition, acquiring of images, potentially with different modalities (microscopy, radiography, fluorescence etc. as highlighted in section 3.2). In the following subsections, we describe conventional/ML based and deep learning based computer vision techniques for automatic or semi-automatic image analysis of fish bio-images in the context of morphometric and phenotype studies.

3.3.1 Conventional Machine Learning methods and algorithms

For small datasets, which are common in biomedical imaging, traditional ML methods, as described in [202], were frequently used in morphometric studies of zebrafish larvae before the advent of DL. In this paper authors addressed the problem of landmark detection in bioimages of zebrafish larvae with a supervised learning approach called extremely randomized trees (ET) algorithm [62]. In this work, each pixel is treated as observation in a large training sample of pixels extracted either from the close neighborhood of the landmark or some randomly chosen positions from the training images. In the classification setting, a separate model is trained for each landmark to predict whether each image pixel belongs to a landmark or not. In regression setting, also a separate model is trained for each landmark in order to predict the Euclidean distance of each pixel from the target landmark. The method is applied to multi-resolution input features of the pixels at different scales and distances. Training has been done using an ensemble of fully-grown decision trees without bootstrapping. The method is also tested on other two publicly available biomedical datasets namely CEPHA [210] and DROSO [173]. A similar supervised machine learning approach using extremely randomized trees for automatically classifying brightfield images of wildtype zebrafish embryos based on their defects has been discussed in [92]. In this work, authors described a machine learning-based image classification algorithm that involves extracting dense random subwindows from images, describing them using raw pixel values, and then classifying these subwindows using ensembles of extremely randomized trees. Finally, the classifications of these subwindows are combined to determine the classification of the entire image. Specifically, the method begins by extracting 1000 subwindows of random sizes and positions from each training image. The sizes of these subwindows are controlled by parameters defining their minimum and maximum sizes relative to the total image size. These subwindows are then resized to a fixed size of 32×32 pixels and described by their raw pixel intensity values in a normalized red-green-blue (TRGB) color space, where pixel value distributions are normalized within each subwindow by channel (subtracting the mean and dividing by the standard deviation). Thereafter, an image of the zebrafish embryo is classified according to its defects by joint exploitation of these subwindows. The method can be employed in two modes: first, where subwindows and, consequently, images are directly classified; or second, where image features, based on the frequencies of subwindows in terminal nodes are classified using a linear SVM method, also trained on the training set. Another approach to automatically classify the absence or presence of malformation in the spine of the medakafish embryo (see in

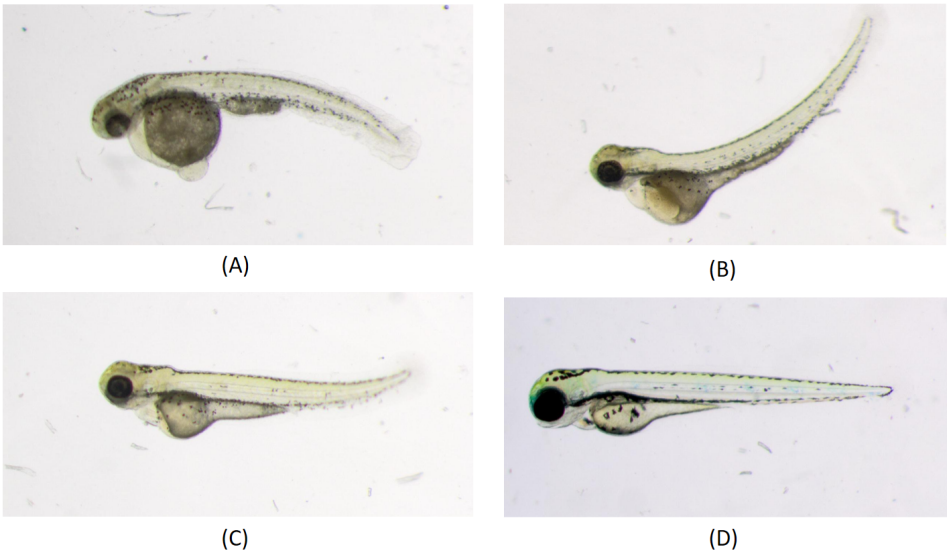

Figure 3.2: Different phenotypes in zebrafish tail. Larvae were imaged live under a dissecting microscope under transmitted light illumination: (**A**) Downward curved tail; (**B**) Upward curved tail; (**C**) Short tail; (**D**) Normal phenotype (source: [92]).

Figure 3.3) is discussed in [61]. In this work, a dataset of 2D high resolution microscopic images of medakafish is used. Features extraction is performed firstly by segmenting the embryo from the images. Since most of the malformations are characterize by abnormal spin curvature, features such as dimensions, curvature angle is extracted. These features are then fed to Random Forest Classifier (RFC) for training. Since feature characterization depends upon the geometry of the skeleton representation of embryos, authors admitted that their methods could not be applicable where tail of the deformed embryo makes a hook shape (see subfigure *'f'* in Figure 3.3), hence not universal to any type of malformation or with high degree of severity in the deformed tail.

When dealing with image data, SVM and boosting techniques are considered alternative choices for classification problems. One such approach is described in [87] for automatic quantification of zebrafish tail deformation. This method is based on estimating the tail curvature of the zebrafish by measuring the partial segments of tails using refined medial representations (RMR), then subsequently fuse these segments in order to get the complete tails. Two data sets containing 67 and 72 images of well plates with 5 fishes per well plates have been used. Authors tested four classifiers namely Naive Bayes, SVM with linear kernel, SVM with radial basis function (RBF) kernel and Adaboost classifier. They reported that SVM with RBF kernel achieved the highest accuracy i.e. 95% per well and 91% per fish. They also reported that some of the misclassifications

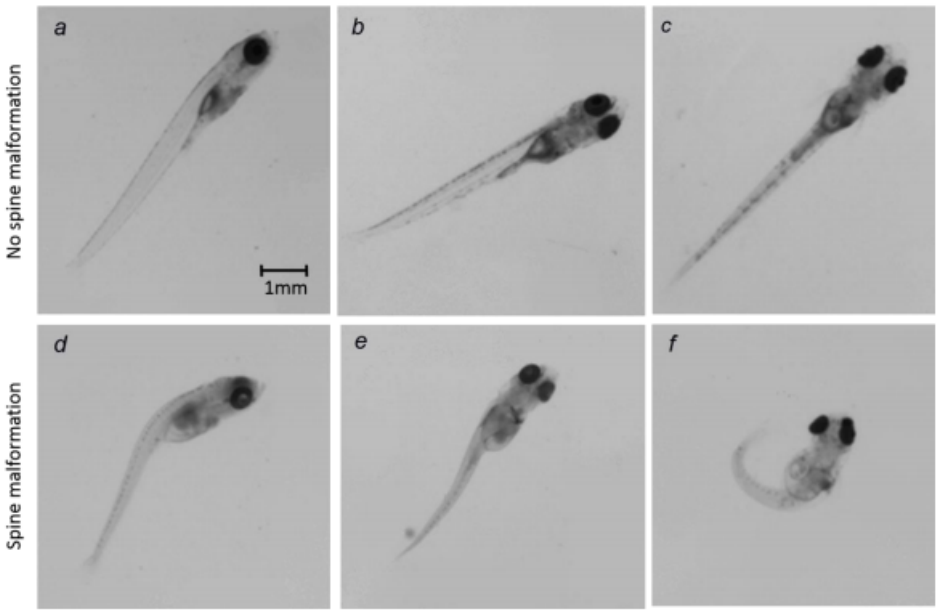

Figure 3.3: Images of 9 dpf Medaka alevins. a to c: healthy alevins shown in lateral view in a, three-quarters view in b and dorsal view in c. d to f: alevins showing different types of spine malformations, d being a major spine malformation (lateral view), e, a slight "S-shaped" malformation (three quarter view) and f a hook-shaped alevin (dorsal view) (source: [61]).

.

were due to debris in the micro-plate wells. In another approach, an ensemble based machine learning method is used to classify compounds that evaluate the behavioral phenotype assays and quantify the screen performance of a zebrafish [135]. A dataset of two sets of compounds comprising 16 quality-controlled compounds and a reference set of 648 known central nervous system ligands are chosen for training and random forest classifier is trained to discriminate between compound induced phenotypes. Many biological studies involve movement behavior analysis and its relevance to ecological studies, toxicology research, or investigations into the effects of various environmental factors on fish behavior. In [106], the authors used a decision tree approach to analyze and categorize medaka fish movement patterns. These patterns describe the process of collecting movement data from medaka fish and using this data to identify specific patterns or behaviors. These patterns could include swimming speeds, directional changes, or other locomotion-related traits that might be the indicators of some morphological changes in fish body.

Supervised machine learning algorithms work well when the data is well annotated.

In contrast, a conventional template matching based approaches do not need any labelling of data and still able to perform well for object detection and segmentation tasks. One such method for detecting and segmenting the head from the microscopy bio-images of zebrafish and medaka is discussed in [192]. This work discusses the multi-template matching approach, which involves using multiple templates (reference patterns) to detect and locate objects of interest within microscopy images of zebrafish and medaka. Template matching conducts the search by moving the template across the image, essentially identifying objects with orientations similar to the template. To enhance the capability for object detection, algorithm permits the input of multiple templates for the search process. This includes the option to include additional perspectives and scales of objects. Additionally, users can modify initial templates by selecting various flipping and rotation options through the plugin interface. To monitor the physical activities and swim pattern of a model fish larvae, statistical analysis and tracking of multiple zebrafish larvae is also performed using gaussian mixture models [114, 213]. In [214], a method is developed to detect and track multiple zebrafish larvae using adaptive gaussian mixture models and Kuhn-Munkres algorithm. For detection and segmentation, an exponentially decaying factor is used to update the model parameters recursively and detection period of larvae is extended if no movement happens for a certain period of time in subsequent frames. Identity assignment and association for each individual larvae are accomplished in consecutive frames using Kuhn-Munkers algorithm [102].

3.3.2 Deep learning based analysis methods and algorithms

In recent years, considerable progress has been obtained in the field of AI development. In particular, deep learning techniques are used for automatic image analysis in biomedical sciences and are becoming the predominant choice in various morphometric and phenotype studies [113, 182]. For single-cell phenotype assays that require gathering complex data at the cellular or sub-cellular level to discriminate features linked to cellular shape, protein localization and intracellular movement, classifying phenotypes using deep-learning methods have proven to be more effective than conventional approaches [53, 191]. A similar deep learning based work is carried out in [51], where the goal is to investigates the application of deep learning techniques for automating the process of cell detection in wide-field microscopy images of zebrafish. The research explores various CNN architectures and training methodologies to optimize performance for identifying cells in complex biological images, aiding the biologists for studying cellular processes and organisms.

The diverse set of observable traits or phenotypes that researchers monitor during embryonic development include morphological changes, cellular behaviors, or other features. These different zebrafish larvae phenotypes are useful for studying the environmental influence on embryo development. The limited availability of annotated data makes it challenging to classify these phenotype traits, as the differences between them can be subtle and ambiguous. In the work of [169], the authors device a two-tier deep learning based pipeline where the CNN model with compressed separable convolution kernels is adopted to address the overfitting issue caused by insufficient training data. Authors report an averaged accuracy of 90% for all the phenotypes and maximum accuracy of 100% for some phenotypes (e.g., dead and chorion), thereby improving the accuracy to 22% against the baseline in [92]. This study offers an effective deep-learning solution for classifying difficult zebrafish larvae phenotypes based on very limited training data. A similar problem of phenotype classification of zebrafish larvae in high-throughput screening using end-to-end deep-learning approach is described in [88]. In this study, the authors tackle the challenge of categorizing morphological alterations in zebrafish found in multi-fish wells, which often have fish overlapping with one another. Assessing the stage, either as a component of a mutant phenotype or induced by treatment, is essential for analyzing morphological changes and developmental delays in zebrafish embryos within a specific time frame. However, the detection and quantification of these delays is often achieved through manual observation, which is both time-consuming and subjective. The work in [93], presents a method for automatically determining the developmental stages of zebrafish embryos using deep learning techniques. In this study, the authors introduce KimmelNet, a deep learning-based pretrained model that is a simplified version of AlexNet [100], that can analyze 2D bright-field microscopy images and accurately predict the age of zebrafish embryos. By leveraging a convolutional neural network (CNN) architecture, the model achieves high accuracy in staging embryos, which traditionally requires manual and time-consuming analysis by experts. The approach has the potential to streamline research in developmental biology, improving the efficiency and consistency of embryo staging.

An advantage of employing deep learning based CNNs is the ability to transfer learned features from one type of task to another (see Section 2.5 about transfer learning), which proves beneficial in situations where there is a shortage of well-annotated data. In [199], a pretrained convolutional neural network (CNN) called VGG-16 [171] is fine-tuned for the task of automated classification of various phenotypical changes induced by toxic substance in zebrafish embryos. During fine tuning, initial layers are freezed and

later layers of the VGG-16 are modified to classify the the 11 observable phenotype traits of zebrafish larvae from the dataset produced by [92]. Environmental and genetic factors that influences the process of embryo development, also encourage the biologists to study the zebrafish eggs in their experiments. Due to the high throughput of microscopic imaging, automated analysis of zebrafish egg microscopic images is highly demanded. However, conventional ML algorithms for zebrafish egg image analysis suffer from the problems of small imbalanced training dataset and subtle inter-class differences. To handle these bottlenecks, a transfer learning with data augmentation based deep learning approach is proposed in [170]. In this work, VGG-16, pretrained on imagenet is used as backbone for the task of automatic classification of weather the egg is fertilized or unfertilized. This study expands the application of deep transfer learning techniques to classify zebrafish egg phenotypes, assisting the biologists in automated analysis of bright-field microscopic images.

As pointed out in the works of [99, 160, 201], genetic inheritance is considered as important factor when studying bone related abnormalities in human beings. Genetic skeletal disorders (GSDs) represent a varied and complex group of rare bone growth abnormalities resulting from disturbances in skeletal development processes, growth pathways, and homeostasis. These disruptions stem from mutations in various genes essential for skeletal system development [200]. Gene editing is one of the effective methods applied to model animals to see the effects of modified genes on the skeletal disorders of the model animal. In [136], a U-Net based image segmentation protocol is proposed to quantify phenotypes of altered renal, neural and craniofacial development in *Xenopus* mutant zebrafish embryos in comparison with normal variability using images of various modalities. These algorithms increase the sensitivity and throughput of evaluating developmental malformations caused by chemical or genetic disruption. Furthermore, authors also provide a library of pre-trained networks and detailed instructions for applying deep learning to the reader's own datasets. Segmentation techniques are needed to study early heart development in model animals by measuring changes in heart chamber volume. Accurate segmentation of the complex shape of the ventricles after trabeculation (transformation from early sponge like structure to smooth and solid shape) begins is essential for analyzing heart function. However the time-consuming task of manually segmenting the light-sheet fluorescent microscopy (LSFM) bioimages is infeasible when processing high axial resolution data, as the number of images required is very large. Recently, deep learning-based bioimage segmentation methods have shown accurate segmentation of zebrafish hearts during the early stages of ventricular development.

[8, 228].

3.4 Fish image analysis in aquaculture

Aquaculture industry provide ample food for human consumption and due to increasing demand, it is facing pressure from the customers to increase the food supply. A decade ago, fish farmers used manual methods for routine tasks like sorting fish by size, identifying diseased fish, removing deformed or dead fish from healthy ones, and counting the number of fish. These manual methods require significant technical effort and time, due to which fish farmers were experiencing great challenges to run a farm with adequate or optimal supply of the fish food. Nowadays, computer vision based image processing techniques are used in aquaculture industries to overcome the challenges of manual and laborious procedure. Although new in this field, computer vision based image analysis methods are helping the fish farmers by speeding up their routinely tasks and at the same time assisting the technicians and researchers in the aquaculture industry to identifying and classifying the fish disorders/deformities.

3.4.1 Conventional image analysis methods and algorithms

To produce high quality fish, selective breeding programs are one such effort in which genetically and phenotypically superior fish breeds are selected for the reproduction. The most effective approach in the selective breeding involves the consistent collection of individualized phenotype measurements throughout an organism's life cycle [63]. Traditional manual methods for assessing fish growth are typically time-consuming, expensive, and stressful for the animals. Even with the use of anesthesia and proper husbandry practices, measurement remains a stressful event. For both cost and ethical reasons, it should either be non-intrusive (i.e., without removing the fish from the water) or performed as infrequently as possible [89]. In the work of [198], a study has been conducted to emphasize the use of automated image analysis techniques for understanding the individualized growth and population structure of Chinook salmon. In the context of image analysis part, the fish's contour is extracted, and 11 reference points (landmarks) are placed on the contour to measure the body side area, fork length, and body height of the fish. Initially the image analysis is performed using OpenCV's conventional thesholding method but subsequently replaced by MXNet-based [39] deep learning models.

Individual fish identification is crucial when it comes to tracking the fish fish behaviour. Since fishes are grown and reared in populations, individual fish tracking become a challenging task due to overlapping of fishes in the tank. In [231], authors implemented a conventional computer vision based method to segment the individual fish in the tank. In the method, firstly the shape factor is employed to identify image overlaps. Subsequently, corner points are extracted using the curvature scale space algorithm, and a skeleton is generated using the improved Zhang-Suen thinning algorithm. Finally, the method identifies intersecting points and effectively segments the overlapped regions within the images. Authors also compared the method with other traditional computer vision based methods such as watershed and Liu's method and reported better results. Another similar traditional computer vision based image processing technique is employed in [46] to automatically measure the length of tilapia fish. In this work, image segmentation is performed using image processing based morphological operation such as dilation and erosion is applied after converting the image into binary image using thresholding method.

Aquaculturists (or fish farmers) perform fish quality checks at regular intervals in their fish farms. Various factors such as treatment, handling, storage, exposure to pollutants, and climate variations, significantly influence fish quality. Distinctions in quality are observed between fish raised in unpolluted freshwater environments and those subjected to polluted or pesticide-affected waters. Pesticides pose a substantial risk to both fish quality and human health. Detecting and identifying pesticide contamination in fish pose a great challenge while using traditional intrusive checks that can lead to high stress level in the fish. In the work of [167], a non-intrusive computer vision based approach is applied to classify whether the fish is contaminated with pollutants or healthy for human consumption. In this approach, pupil and eye of the fish are selected as region-of-interest (ROI) to extract the discriminative features for the classification. First the ROI is segmented using traditional thresholding methods. Then first order statistical analysis is performed to extract features such as mean, standard deviation and variance of the 'S' channel of HSV colour space. On these features, different types of classifiers such as Support Vector Machine (SVM), Artificial Neural Network (ANN), Naive Bays Classifier and Random Forest (RF) are tested. Among all the classifers, authors reported better results with RF classifier. Object identification or locating the region of interest (ROI) from a digital image is generally performed using computer vision based image segmentation methods. In [221], an improved k-means clustering algorithms is applied for the extraction of contours of the fish followed by morphological operations such as

dilation, erosion, opening, closing to separate the fish boundaries from the background. The authors reported that algorithm offers improvements over the traditional K-means approach and focuses on effectively separating fish from their background in images, which is crucial for various applications in aquatic research and fisheries management. Certain machine learning based image segmentation method are useful in fish counting as well. In the work of [229] the study involves the segmentation of fish-connected regions in top-view fish images, obtained through morphological image progressing operation. Subsequently, four types of image features are extracted from each of these fish-connected regions while removing the redundant features with principal component analysis (PCA). Fish counting is then executed by applying image density grading using threshold method based on the area of the connected area. This approach divides fish images into several sub-images, each containing connected areas, and performs density grading on each sub-image. This helps rectify the imbalance in the dataset of fish-connected area sub-images, resulting in more precise and consistent fish counting. Finally fish counting is performed using a fish-number prediction model, based on BPNN (Backpropagation Neural Network) for the connected-area dataset at various density levels. The trained model was then used to determine the fish count within each connected-area image. The local counts is determine by combining each fish-connected area image.

Identification of disease in the farmed fish is a challenging task. Manual invasive method involves picking the fish from the water and inspecting it by the expert for the potential disease identification. These intrusive methods are not only tedious and time consuming and require technical expertise but also put the fish under stressful conditions. In [7] an ML based method is devised to automatically diagnose and identify infection in the salmon fish. The approach is divided into two main components; 1. In the initial phase, image pre-processing techniques is employed to reduce noise and enhance image quality, 2. the second phase involves extracting relevant features to facilitate the classification of diseases using the Support Vector Machine (SVM) algorithm with kernel function. In [123], an artificial neural network (ANN) is used after feature extraction with the FAST (Features from Accelerated Segment Test) algorithm. The proposed method begins with image preprocessing, where images are converted to grayscale, contrast is enhanced, noise is removed, and segmentation is performed. Next, discriminative features are extracted using the FAST algorithm, and Principal Component Analysis (PCA) is applied to reduce dimensionality for improved prediction. Finally, an ANN classifier is employed on these features to identify diseases in the fish images.

3.4.2 Deep learning based methods and algorithms

Conventional ML methods in the aquaculture research (as outlined in Section 3.4.1) are being used for more than a decade, in which manually crafted features require a large number of human effort and introduce additional uncertainty factors. Recently computer vision based deep learning methods are becoming popular due its performance and easy to use traits. Deep learning methods have the capability to automatically discriminate both high and low level features from dataset itself, allowing for the detection of subtle fish characteristics within images. It remains resilient against variations in lighting, positioning, and orientation, rendering it well-suited for computer vision modeling [220]. Although new to aquaculture, CNNs are now getting helpful in various aquaculture activities such as measuring the length of the fish, landmark detection, fish body part segmentation.

In [194], a deep learning based method is proposed to automatically measure the length and weight of Meagre fish in a non-invasive manner. In this work, fish stereo images are first fed into a deep learning based 'You Look Only Once' (YOLO-v4) [152] object detector to draw the bounding boxes over each individual fish. Subsequently, each bounding box is utilized to extract the individual fish image, which is then processed through pre-trained ResNet-101, a CNN optimized for image recognition. The last layer of ResNet-101 is modified to detect two landmark positions on the fish namely the snout tip and the base of the middle caudal rays. The landmark detection algorithm measures the fish length in pixels by counting the distance between the two landmark points. Finally, the pixel-based length was converted to centimeters using translation information derived from the calibration phase involving chessboard target images. Similar approach using advanced version of YOLO (YOLO-v5) is applied in [125] for the measurement of the fish length combined with stereo-BRUVS calibration method, which uses calibration cubes to ensure precision within a few millimetres in calculated lengths. YOLO object detection combined with DeepSORT algorithm [217] is used for the fish tracking and behaviour analysis in [83]. In this work YOLO-v5 is used to detect fish in the image and drawing the bounding box over it, while DeepSORT algorithm is used for fish tracking. In DeepSORT algorithm, first the previously predicted trajectory is estimated using Kalman filter module then Hungarian algorithm is applied to assess the level of correspondence between the current frame's detected result and the predicted track. Subsequently, any inaccurate tracks are removed to finalize fish tracking, and the accurate tracks are updated through the Kalman Filter's update module.

Selective breeding aimed at enhancing swimming abilities in fish might lead to

morphological changes of their offspring. These changes, while benefiting industrial productivity, also have implications for the welfare of the animals involved. To measure the morphological changes in the body of the fish, shape analysis is performed by placing important landmarks on the fish body in the images. Manually marking the landmark points is laborious and requires technical expertise. In order to determine 10 morphological traits correlated with swimming performance in 'juvenile large yellow croaker', a deep learning based CNN called high-resolution network (HRNet) is proposed in [227] to automatically locate the anatomical landmarks points in bioimages of fish. To get the scaling relationship from the pixel distance on the image to the physical distance in the metric system, pixel length of the reference solid line on the image is detected and divided it by the physical distance of the reference solid line of 10cm. In this paper, the threshold segmentation method is used to detect the pixel length of the reference solid line. First, RGB image is converted into gray-scale. Then, an appropriate threshold is set to binarize the grays-cale image to segment the reference solid line. Finally, pixel length of the reference solid line is computed after segmentation.

As discussed previously, fish disease is a prime concern that lead to increasing deaths in the fishes and ultimately the potential reason for the economic loss. In the work of [209], authors compared several CNN architectures such as AlexNet, ResNet18, ResNet50, ResNet101 for the classification of fish diseases.

3.5 Fish bioimage analysis tools

After a comprehensive literature review of various conventional, machine learning (ML), and deep learning (DL) methods used in biomedical and aquaculture research, this section focuses on bioimage analysis tools that feature interactive user interfaces and integrate ML or DL models as software packages.

3.5.1 EmbryoNet

EmbryoNet [30] is a deep learning based software tool to identify the phenotype defects in the embryonic stage of the zebrafish. The aim of this approach is to bridge the gap between observed phenotypic traits in embryos and the underlying molecular signaling pathways responsible for those traits. These diverse set of observable traits or phenotypes that researchers monitor during embryonic development include morphological changes, cellular behaviors, or other features. In this work, a deep learning

based convolutional neural network (CNN) called *'EmbryoNet'* is trained to classify the phenotypic defects caused by loss of function of the seven major signaling pathways relevant for vertebrate development using zebrafish signaling mutants combined with a model of time-dependent developmental trajectories (see Figure 3.4). EmbryoNet is a modified version of 'ResNet18' in which time stamp channel as additional input dimension is added, thus feeding four instead of three channels, and by replacing the last classification layer with the current classification layer. EmbryoNet is trained on more than 2 million images, comprising thousands of trajectories of normally developing and signaling-defective zebrafish embryos. The Authors also apply the model to other fish species such as medaka (Oryzias latipes) and three-spined stickleback (Gasterosteus aculeatus) to test the generalizability of the approach. The approach enhances the ability to predict developmental outcomes and understand the mechanisms driving embryogenesis, making it a valuable resource for developmental biology and genetics research.

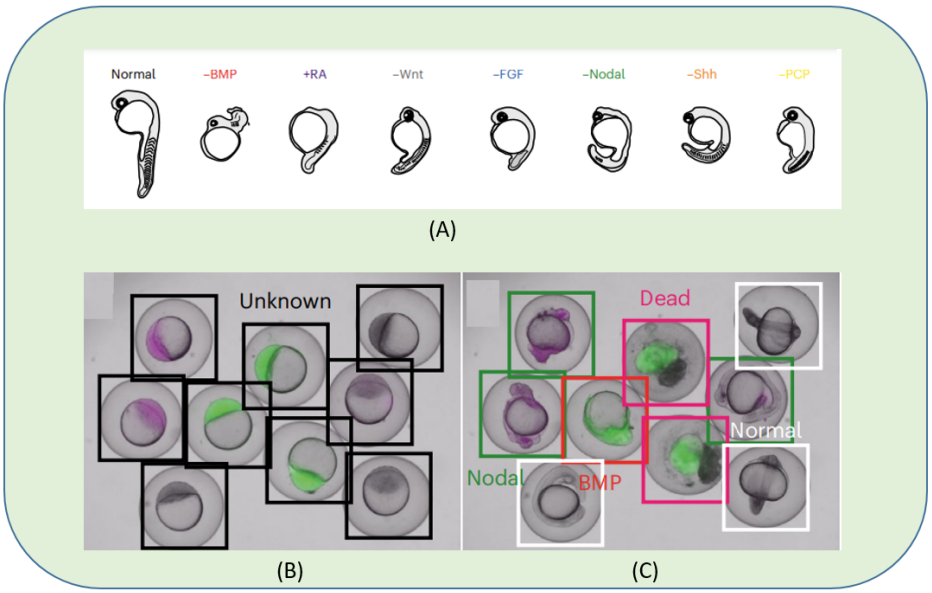

Figure 3.4: Schematic representation of 7 types of embryo phenotypes (A). Classification using 'EmbryoNet' at different stages: at sphere stage (B) and at 24 dpf (C). (source: [30])

3.5.2 QuantiFish

Quantifish [179] is a software tool, designed to measure and analyze the spread of pathological conditions in zebrafish larvae. It is designed to quantify pathogen or bacterial load

(infection) by measuring the spatial distribution of bacterial foci, specifically calculating the number of bacteria per macrophage in zebrafish larvae. This represents the disease severity by measuring four parameters, namely the number of fluorescent bacterial foci that are responsible for 50% of the total fluorescence, the number of predefined grid zones that contain the centre point of a bacterial focus, third the area of a polygon containing the centre points of all foci and the maximum distance between any two foci. The authors in the paper stated that the total bacterial load (integrated fluorescence intensity) and the numbers of separate foci of bacteria detected using QuantiFish were significantly higher in fish with more widely disseminated infection. The approach allows researchers to systematically study disease dissemination, aiding in the understanding of disease mechanisms and the evaluation of potential treatments. The graphical user interface (GUI) of 'QuantiFish' software is depicted in Figure 3.5

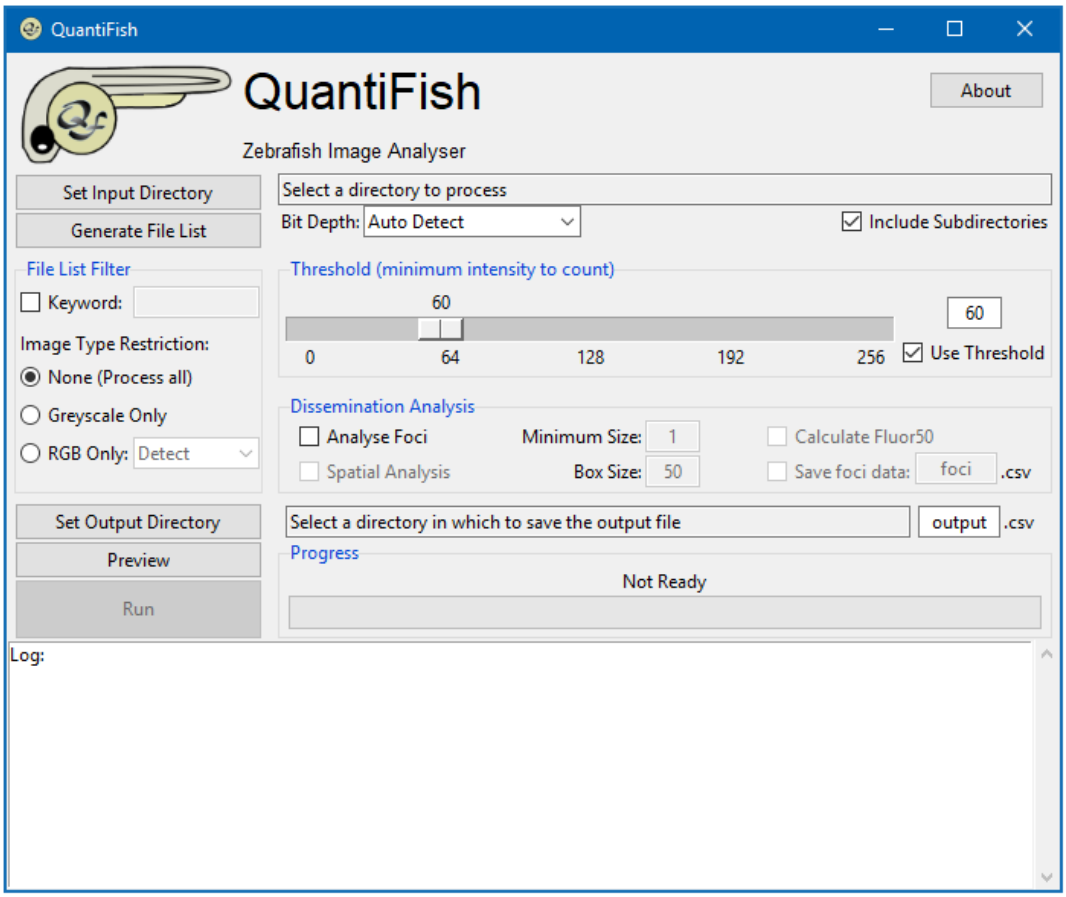

Figure 3.5: Depiction of GUI of'QuantiFish' software tool for automated quantification of fluorescent intensity in microscopy images of zebrafish. (source: [179])

3.5.3 ZF-AutoML

ZF-AutoML [164] is a tool that is devised to detect macrophages anomalies from fluorescence-labelled zebrafish. The tool is based on Google's AutoML cloud platform [23] which is used to train and evaluate custom machine learning models for detection and classification tasks. The method automates the process of identifying irregularities in zebrafish images, making it accessible even to users with minimal machine learning expertise. ZF-AutoML is developed for the classification of normal phenotype (control) and abnormal phenotypes (sorafenib-treated and wounded fishes for the angiogenesis and macrophage experiments, respectively). The tool streamlines classification task by leveraging Google's AutoMl based advanced ML algorithms to analyze fluorescence patterns, enhancing the efficiency and accuracy of identifying developmental or pathological abnormalities. This tool is particularly useful for researchers in developmental biology and toxicology, enabling high-throughput and reliable analysis of zebrafish models. Figure 3.6 shows the schematic representation of ZF-AutoML machine learning process.

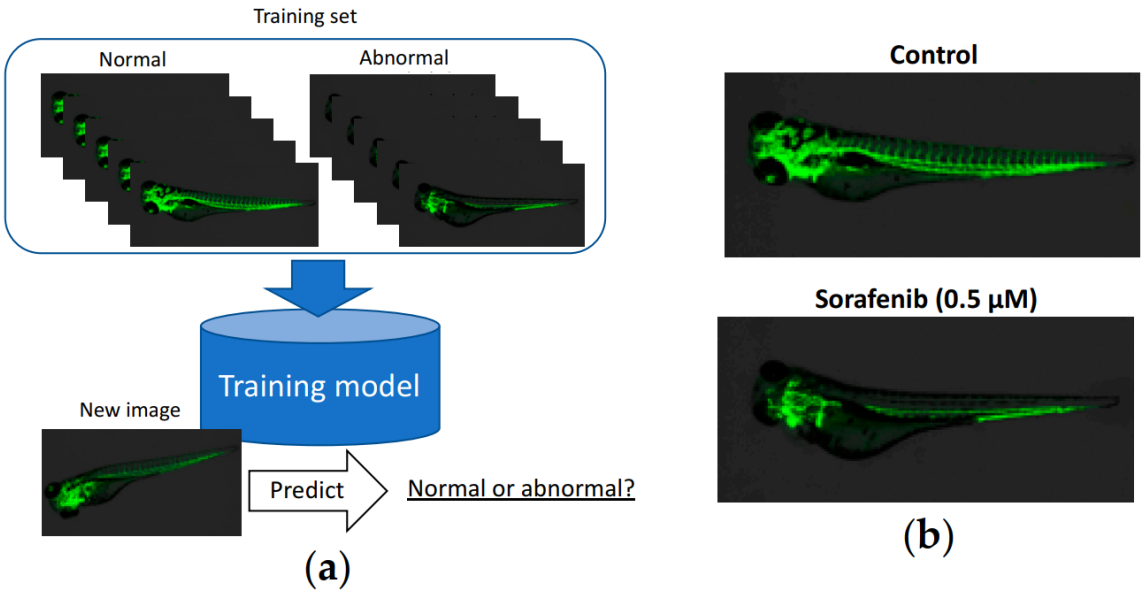

Figure 3.6: (a) Schematic representation of AutoML machine learning process. (b) Images of Tg (kdrl:EGFP) a zebrafish strain at 96 h-post fertilization (dpf) with or without sorafenib (0.5 µM). Sorafenib treatment was started at 24 hpf. The green color indicates vasculature. (source: [164])

3.5.4 ZFTool

ZFTool [33] is a software tool developed for automating the process of quantifying the growth and progression of cancer cell masses within zebrafish embryos. The tool enables researchers to track and measure tumor development over time with high precision, facilitating the study of cancer progression in a non-invasive manner. The method starts by injecting cancerous cells (HTC116 cell line) into the embryo's yolk, followed by a 72-hour incubation period to monitor cell proliferation. Each embryo is photographed at two time points: immediately after injection (0 hours post-injection, hpi) and after 72 hours (72 hpi). The images obtained were then subjected to analysis using the ZFTool software, with a focus on the green channel image. The aim of ZFTool is to quantifying cancer mass evolution over time by measuring the number and mean value of GFP (green fluorescent protein) pixels. This measurement is conducted by comparing images taken at 0 hpi with those taken at 24, 48, or 72 hpi, depending on the specific experiment. ZFTool removes the zebrafish autofluorescence through the computation of the area with varying intensity thresholds. It does so by automatically calculating the auto-fluorescence threshold for both the initial (0 hpi) and subsequent time points (24, 48, or 72 hpi) and establishing a baseline threshold to eliminate auto-fluorescence from zebrafish embryos. It quantifies the area occupied by marked cells (GFP) and their intensity through threshold-based segmentation, providing a measure of proliferation known as the proliferation index, which reflects the evolution of cancer mass in zebrafish. Figure 3.7 show the segmentation over a sample image.

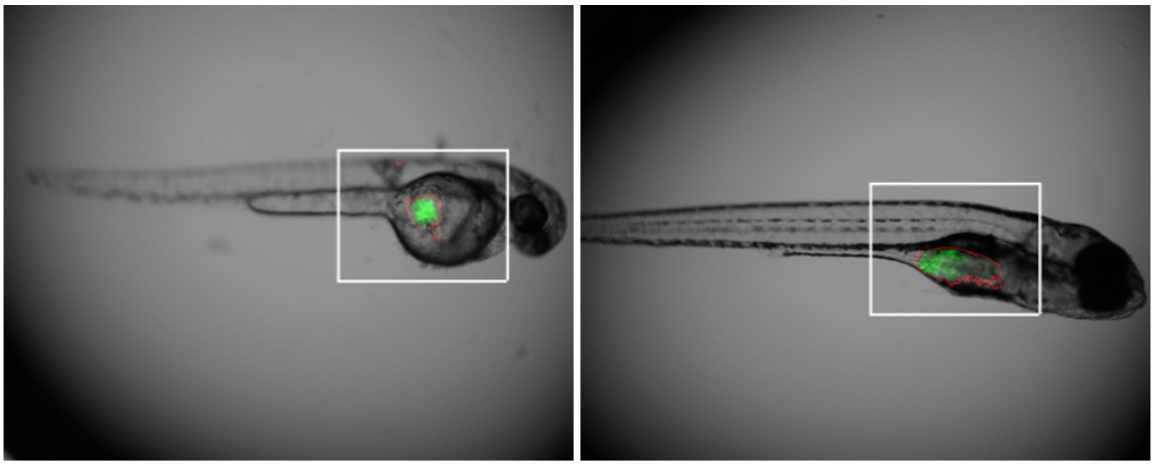

Figure 3.7: Depiction of segmentation over a characteristic image (zebrafish at 0 hpi and 72 hpi) where the GFP value and the contour image are overlaid in green and red, respectively. The white rectangle is the region of interest (source: [33])

3.5.5 ZF-Mapper

ZF-Mapper [219] is a method to quantify the fluorescent intensities of the pixels in the microscopic zebrafish images. The authors tested it with macrophage-specific enhanced green fluorescent protein called EGFP and cancer cell xenograft, implanted in the body of the zebrafish embryos. In the first experiment, the total fluorescence of zebrafish expressing macrophage-specific EGPF at the developmental stage of 2-6 dpf as shown in Figure 3.8 is quantified and results are compared with conventional softwares such as ImageJ for its reliability. In the second experiment, the cancer xenograft fluorescent images of the zebrafish implanted with melanoma cancer cells was analysed. In this experiment, they found that the intensity of the fluorescent regions in the body of zebrafish has increased from day 2 to day 6 which highlights the increment in the number of cancer cells in the body of zebrafish over the period of time. ZF-Mapper is designed to be easy to use, allowing for the efficient processing and quantification of fluorescence data without the need for complex or expensive software. The tool is particularly useful for studies involving gene expression, protein localization, and other fluorescence-based experiments in zebrafish analysis.

3.5.6 ZebraZoom

ZebraZoom [131] is a method for automatically analyzing zebrafish behavior by tracking their movement within a well and identifying the maneuvers performed during episodic movements. In this work, the authors categorized these maneuvers into three types: slow forward swim, routine turn, and escape. They modeled the sequence of maneuvers as a Markov chain between the two events. To track the core positions of the larvae, the background image is first subtracted to create binary masks, and then the image is eroded to determine the core and head direction of the larvae. A bending angle at the tip is defined that separates the body axis from the line connecting the core and the tip of the tail (see part C in Figure 3.9). For categorization, multiclass Support Vector Classifier (SVC) with linear kernel was chosen. To track the full body position over multiple time scale, the core that includes the head area and swim bladder and tail positions are measured simultaneously for multiple larvae and global parameters such as tail bending angle, mid-line position of tail and position of its core and head axis are extracted. These global parameters along with some other local parameters such as amplitude of the tail bending angle, instantaneous frequency over time is calculated. Figure 3.9 shows larvae's tail and detection of movements based on the tail-bending angle. ZebraZoom

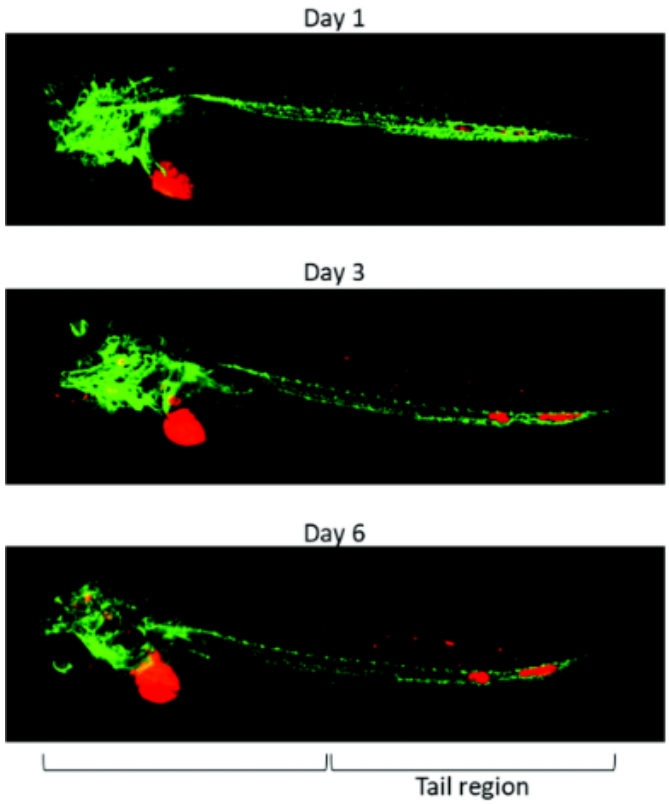

Figure 3.8: Typical images of tdTomato-labeled A375 xenograft zebrafish from day 1 to day 6 after cell implantation. (source: [219])

allows researchers to efficiently monitor and categorize various behavioral patterns in zebrafish, providing a streamlined approach to studying their activity and responses. The software is capable of handling large datasets, making it ideal for research that requires extensive behavioral analysis. By automating the process, ZebraZoom reduces the need for manual observation, increases accuracy, and enhances the ability to conduct large-scale behavioral studies in zebrafish models.

3.5.7 FishInspector

FishInspector [190] was developed to quantify the morphometric defects during developmental toxicity screening in zebrafish embryos. It can analyze large numbers of embryos to detect and quantify morphological abnormalities, which are indicative of developmental toxicity. In this approach morphometric features are extracted and organized in hierarchical manner using length and surface areas from contour information of different parts of the zebrafish candidate. In order to detect certain features, the information

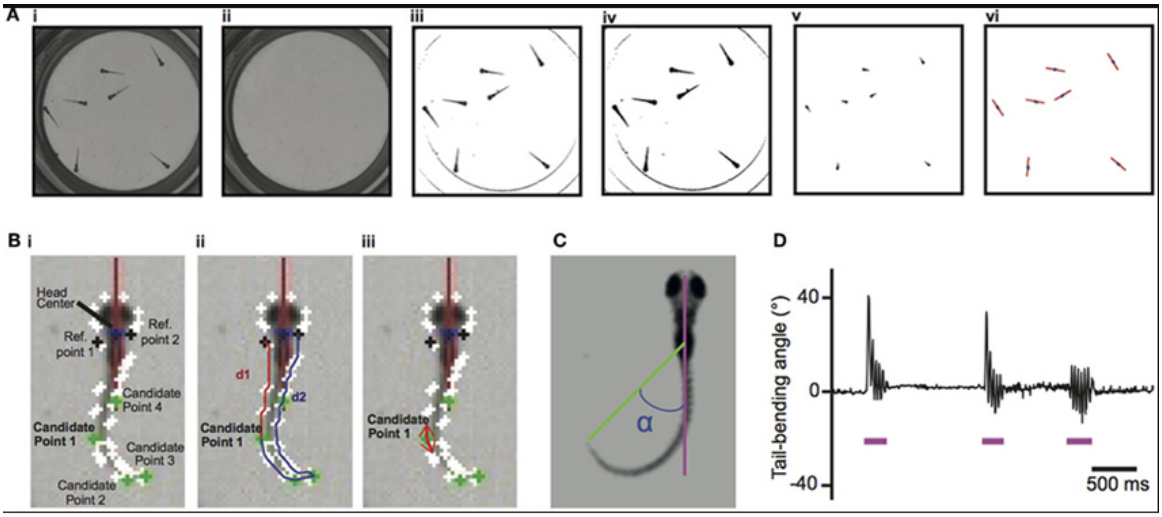

Figure 3.9: Image processing for tracking of larvae's core positions using tail-bending angle. (A) Tracking of the larvae's core positions. (B). Identifying the tip of the tail. (C) Definition of the tail-bending angle (α) separating the body axis (pink) and the line connecting the core and the tip of the tail (green). (D) Example of the tail-bending angle over time with detection of movements indicated by the pink line. (source: [131])

about previously detected features should also be included. Finally, the detected features are the boundary coordinates of the contours of the objects such eyes, head, swim bladder etc. of embryos. Since the detection of some specific features are dependent upon the other features, improper detection of one feature may cause a cascading effect upon other features which may adversely affect the performance of the software tool. The feature detection algorithms employed in the software are based on contour information which is semi-automatic and no self learning algorithm has been used. Authors also specified that, a jaw morphology analysis cannot be performed automatically using this tool hence subject to manual annotation and correction by the user. The system allows for high-throughput screening, making it a valuable tool for environmental and pharmaceutical research, where understanding the developmental impact of various substances is crucial. Figure 3.10 shows the screenshot of FishInspector software showing an image with detected regions of interest (ROIs) for each feature.

3.5.8 Stytra

Stytra [178] is an image analysis tool that enables real-time tracking and quantification of zebrafish behavioral traits, including position, orientation, and eye motion. The

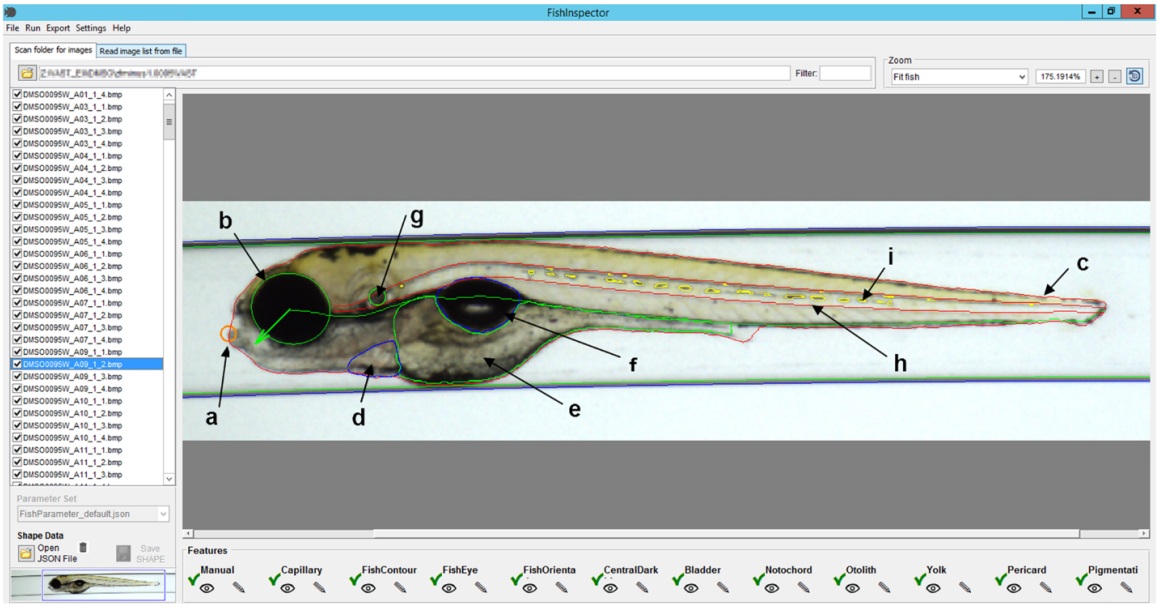

Figure 3.10: Screenshot of the FishInspector graphical user interface showing an image with detected regions of interest (ROIs) for each feature. (source: [190])

experiments utilize both freely swimming and head-restrained zebrafish larvae to examine their behavioral patterns in well-plates. Images or video frames are captured using various supported camera models, such as XIMEA, AVT, and those compatible with OpenCV. For behavior tracking of head restrained fish, curvature of the tail in current position is compared with previous one and tail angle, position and orientation is recorded. For Eye tracking, first elliptical regions of eye balls are segmented and the absolute angle of the major axis of the ellipse is measured as eye angle. For freely swimming fish tracking, the center of mass of the three objects namely two eyes and one swim bladder is extracted from background and taken as center of the fish head. The direction of the tail is measured by searching for the point with largest difference from the background on a circle of half tail radius. The system is highly customizable and user-friendly, making it accessible for a wide range of behavioral studies. By automating complex experimental setups, Stytra enhances the efficiency and accuracy of behavioral research, facilitating advanced studies in neuroscience and behavior. Figure 3.11 shows the screenshot of the user interface of Stytra software tool.

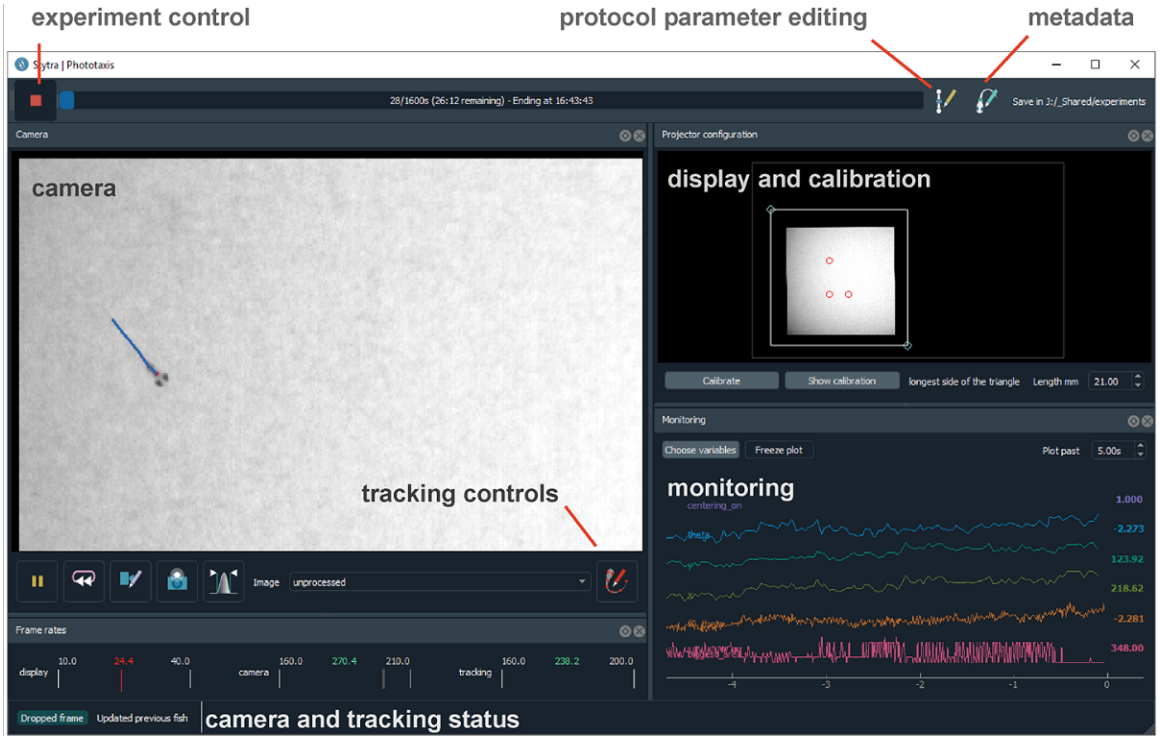

Figure 3.11: Stytra supports a range of behavioral paradigms, offering users a consistent interface for experiment control. The top toolbar manages the experiment's execution, while a camera panel displays tracking results overlaid on the camera image. A calibration panel allows for easy positioning and calibration of the stimulus display, and a monitoring panel provides real-time plots of user-selected experimental variables. (source: [178])

3.5.9 ZebrafishMiner

ZebrafishMiner [153] is an open-source software designed for the interactive analysis of domain-specific fluorescence in zebrafish. Developed as an extension package for the MATLAB Toolbox SciXMiner [129], it provides an image processing pipeline for the automatic quantification of fluorescence in zebrafish embryos and larvae of different ages. The software categorizes fluorescence data into user-defined domains such as tissue, notochord, skin, eye, brain, yolk, and others. A brightfield and a reference fluorescent channel consisting of multiple slices for segmentation is used to assign fluorescent signal. Figure 3.12 shows screenshot of the "Tissue manager“ to colour a tissue in embryo images. For automatic fluorescent evaluation, embryos are detected in brightfield and fluorescent images and compared with reference embryos with known domains. This tool allows researchers to efficiently evaluate and quantify domain-specific fluorescence, facilitating the study of gene expression and protein localization within zebrafish models.

ZebrafishMiner offers a user-friendly interface and advanced visualization features, making it easier to interpret fluorescence data and gain insights into developmental and genetic processes. The software aims to enhance the accuracy and efficiency of fluorescence-based experiments in zebrafish research.

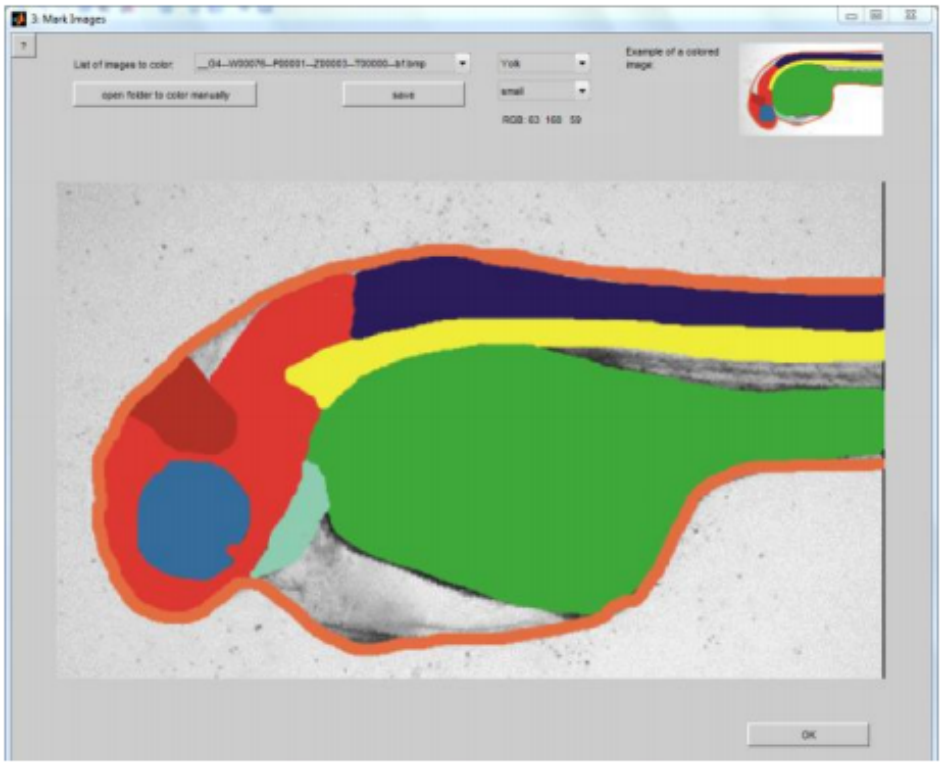

Figure 3.12: Tissue Manager (source: [153])

3.5.10 CNNTracker

CNNTracker [230] presents a method for tracking zebrafish using convolutional neural networks (CNNs). The study develops a CNN-based approach to accurately track the movement and behavior of zebrafish in video recordings. It tracks the individual fish in a group of multiple fishes and maintains the correct identities of each fish after crossing one another or if long time occlusion occurs. In this approach, the head feature maps of each individual fish are first extracted (see Figure 3.13), and head point pairs between successive frames are matched. Fish trajectories are then determined by linking these corresponding head point pairs through inter-frame feature mapping. A collection of head point pairs is created, where the first point is from the previous frame and the

second point is from the current frame, resulting in trajectory segments for each fish. The CNN network is trained using these trajectory segments, allowing it to map and identify individual fish, as each segment corresponds to a unique fish identity. By leveraging deep learning techniques, the method improves tracking precision and robustness compared to traditional tracking methods. This advancement enhances the ability to analyze zebrafish behavior and movement patterns, providing valuable insights for research in developmental biology, neuroscience, and other fields where zebrafish are used as model organisms.

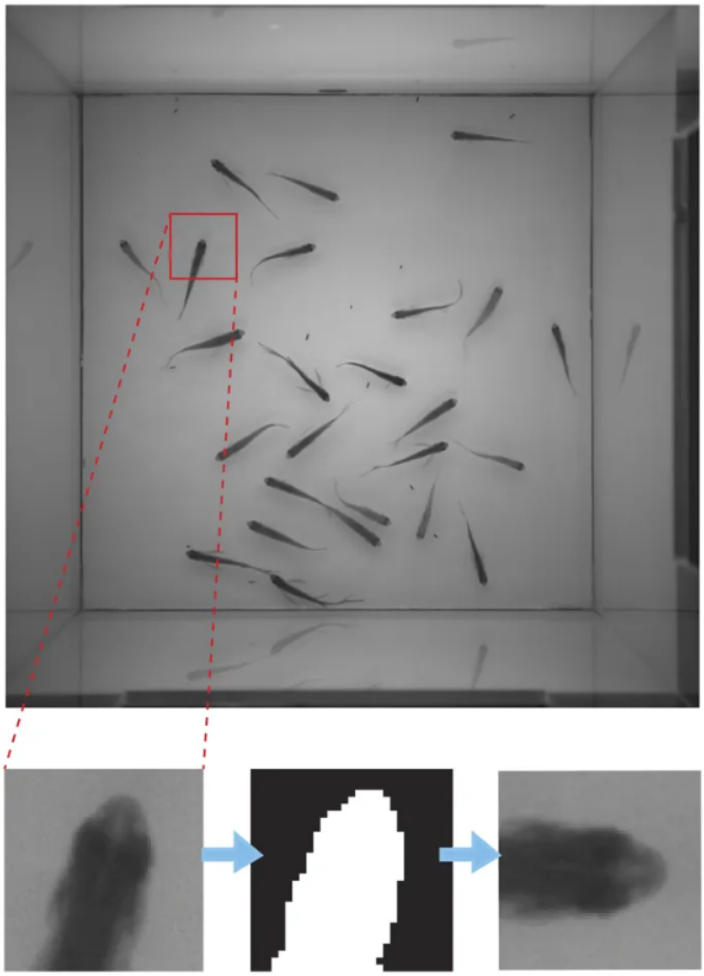

Figure 3.13: The centre of the red square is the detected fish head then image patch is extracted with the head point as the centre of the dashed red line. Head patch is transformed into a binary bitmap and the coordinates of the white pixels in the binary image are extracted. Orientation of the fish head is obtained and rotated to 0 degree to get the final fish head feature map. (source: [230])

3.5.11 DeepLabCut

DeepLabCut [127] is a generic software tool, designed to automatically detect user-defined anatomical landmarks from the video frames of different animals. Its core working mechanism relies on deep learning based pre-trained CNNs, particularly 'ResNet' which acts as the backbone of many CNNs implemented in this tool for tracking the body of the animal in the video. DeepLabCut is particularly useful in behavioral studies across various fields, including neuroscience, biomechanics, and ethology, where accurate and high-throughput tracking of animal movements is essential. One of its strengths is its flexibility, allowing users to train models for tracking different species and body parts, as well as its relatively user-friendly interface compared to traditional tracking softwares. Although, DeepLabCut tool was initially invented for markerless pose estimation in mammals, in [184], the authors explored the possibility of adopting this tool for conducting markerless cardiac physiology assessment in an important aquatic toxicology model of zebrafish. In this work, a high-definition videography of heartbeat data is recorded at a frame rate of 30 frames-per-second (FPS). Next, 20 videos from different individuals are used to perform convolutional neural network training by labeling the heart chamber (ventricle) with eight landmarks. Using ResNet-152, a neural network with 152 convolutional layers with 500,000 iterations, is trained that can track the heart chamber in a real-time manner. Figure 3.14 shows the working of DeepLabCut tool for the application of cardiac physiology assessment in zebrafish.

3.5.12 Icy

Icy [47] is an open-source image analysis software platform that boasts a range of key features designed to facilitate scientific research and bioimaging. It supports multidimensional images, including 2D, 3D, and time-series data, making it ideal for analyzing complex biological samples. Its modular plugin architecture allows users to extend its capabilities with custom or existing plugins for specific analyses. Icy includes a variety of built-in image processing tools for filtering, segmentation, feature extraction, and quantification, complemented by powerful visualization tools that enable interactive displays of data. The user-friendly graphical interface ensures accessibility for researchers, even those with limited programming skills, while scripting capabilities facilitate batch processing for large-scale studies. Additionally, Icy is compatible with various file formats commonly used in microscopy, further enhancing its integration into existing workflows. The active community surrounding Icy provides valuable support through documenta-

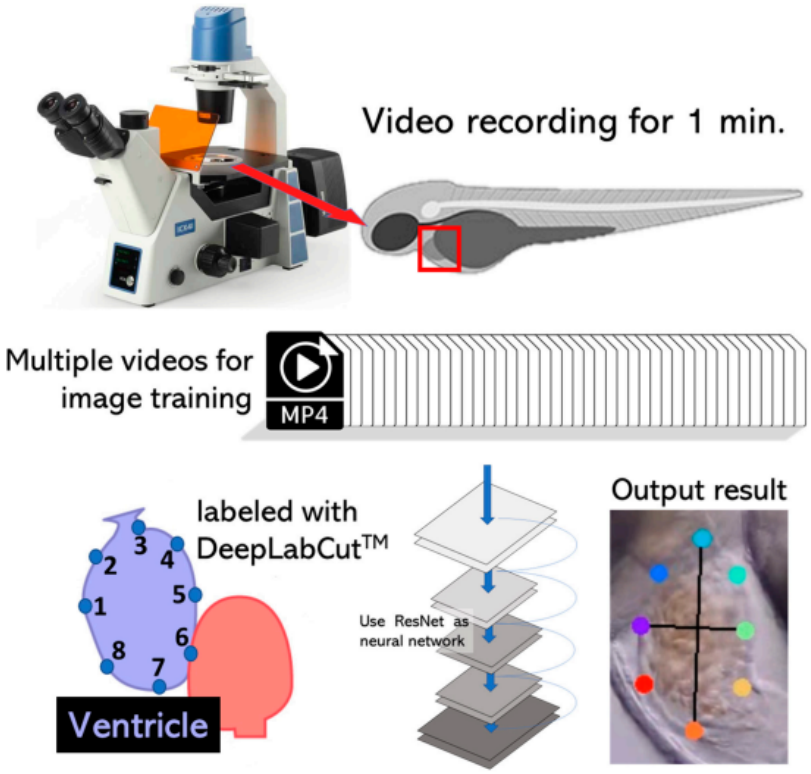

Figure 3.14: The workflow, used to detect and label the heart chamber of zebrafish. On top, the video of the animal model zebrafish is recorded for cardiovascular assessment. Up to 20 videos of a heart beating with a duration of 1 min are collected. The bottom section describes how DeepLabCut performs the training process for dataset and video analysis, resulting in a labeled zebrafish ventricle heart chamber. (source: [184])

tion, tutorials, and forums, making it a versatile choice for image analysis across multiple scientific disciplines. Figure 3.15 shows the applications of Icy software tool to zebrafish research.

3.5.13 Cytomine

Cytomine [124] is a generic open source, web-based software for collaborative analysis of multi-gigapixel imaging data. This software is designed to bring researchers from various fields on one platform to analyse their imaging data in a collaborative and distributive manner. It uses web development methodologies and machine learning in order to readily organize, explore, share, and analyze (semantically and quantitatively) multi-gigapixel imaging data over the internet. Its web based user interface allows researchers, students, collaborators to create, organize, visualize, and edit all data and share projects through

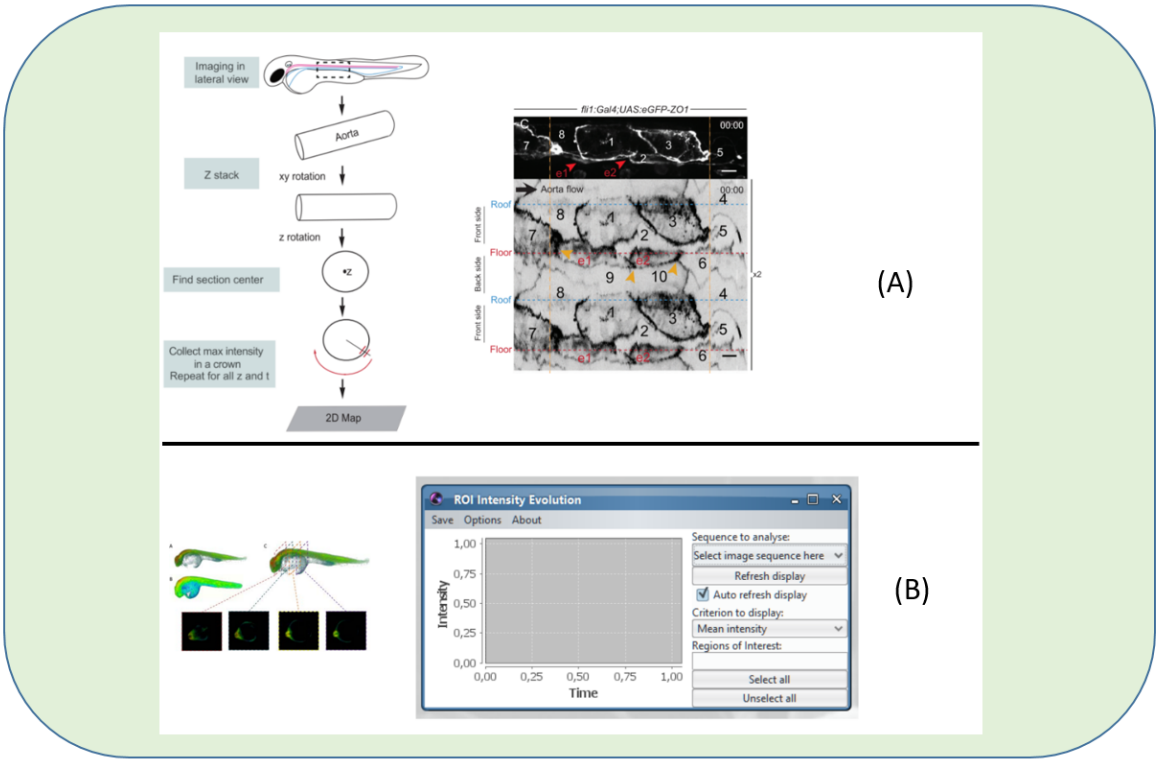

Figure 3.15: Applications of Icy tool in zebrafish research. (A) Quantitative analysis of fluorescence lifetime with the Icy plugin ROI intensity evolution. (B) Unwrapping the aorta tube with the TubeSkinner plugin (source: Icy)

authentication. It has all the advanced image processing tools to analyse, annotate and manipulate multi gigapixels images from multiple sources. It includes designing efficient workflows and creating advanced algorithms, such as deep learning and tree-based machine learning models, to support content-based image retrieval, object detection, recognition, and segmentation in extensive multimodal datasets. Emphasis is placed on reproducible benchmarking of these algorithms across realistic datasets, particularly within the biomedical domain, with a focus on digital pathology and multimodal microscopy. Figure 3.16 shows the zebrafish imaging modalities handled by Cytomine.

3.6 Challenges in bioimage analysis tasks

The development and application of machine learning (ML) and deep learning (DL) models for bioimage analysis in morphometric studies encounter various complex challenges, especially when focused on fish species in aquaculture and biomedical research. These

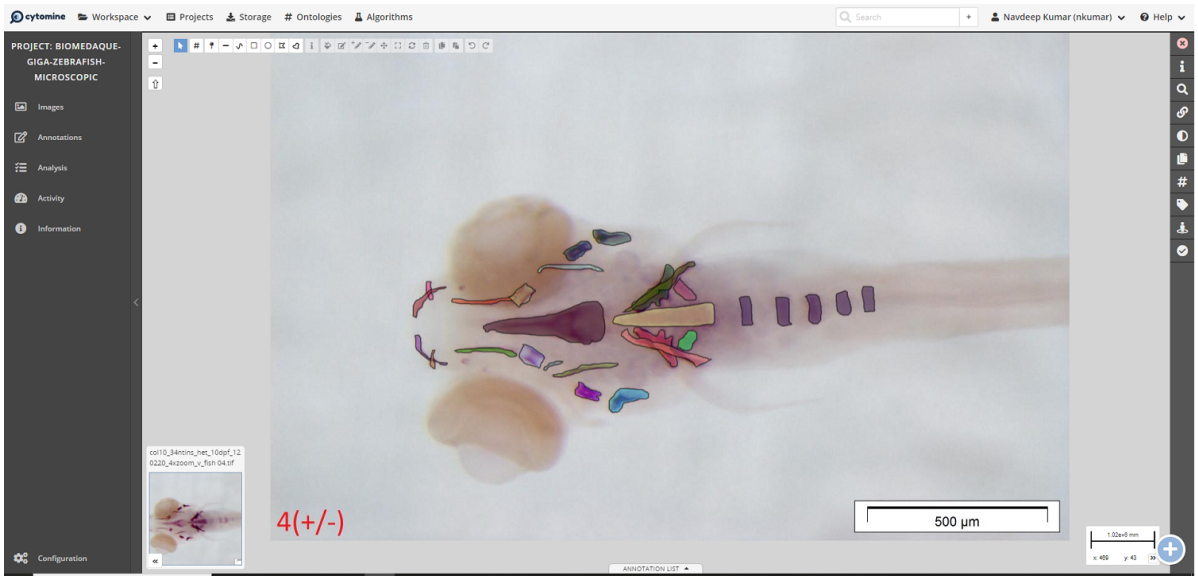

Figure 3.16: Zebrafish imaging screen used in Cytomine for analysis (source:research.cytomine.be)

challenges can be broadly classified into several categories, including issues related to image acquisition, data quality, annotation subjectivity, diversity of modalities, and the availability of common image analysis platforms. Some of the these challenges are outlined in the sections blow:

3.6.1 Image acquisition complexity

Bioimage data in aquaculture and biomedical studies often come from diverse imaging techniques, such as microscopy and radiography, which vary significantly in resolution, lighting, contrast, and noise levels. These variations pose a significant challenge for ML and DL models, which typically require consistent, high-quality data to perform optimally. Noise and variability introduced during image capture, especially under non-standardized conditions, can hinder model accuracy and lead to inconsistent results.

3.6.2 Lack of annotated bioimages

The limited availability of well-annotated bioimage data poses a significant challenge when developing image analysis tools or methods for biomedical research, which rely on annotated datasets for training, tuning, and validation. While natural image datasets are readily accessible on various platforms as open-source resources, bioimage datasets are

typically obtained using costly instruments under controlled laboratory conditions, making them less accessible to the general public. Most bioimage datasets are derived from either patients or model organisms and are often confined to specific labs or researchers. Additionally, these datasets usually contain only a small number of images—ranging from a few hundred to a few thousand—compared to natural image datasets, which can feature millions or even billions of well-annotated images. This limited availability of annotated images often proves insufficient for training deep-learning models, resulting in subpar performance. Consequently, models trained on inadequate image data tend to face difficulties, hindering analysts' ability to utilize them effectively for image analysis tasks.

3.6.3 Class imbalance

Many bioimage datasets, especially those dealing with specific anatomical structures or rare phenotypes, are highly imbalanced. For instance, images where certain developmental features or disease markers are absent may vastly outnumber those where they are present. This class imbalance can lead ML and DL models to become biased towards the majority class, reducing sensitivity to rare but biologically significant features. Handling class imbalance effectively is essential for reliable phenotype classification, segmentation, and structure detection.

3.6.4 Annotation subjectivity and inconsistency

Accurate annotations are critical for training supervised ML and DL models, but these annotations are often subjective, particularly in the biomedical domain where boundaries between structures may be unclear or open to interpretation. Variability in expert annotations can lead to inconsistencies in the training data, complicating model training and evaluation. Furthermore, detailed, high-quality annotations are time-consuming and labor-intensive to produce, and errors or disagreements among annotators may introduce noise that models can inadvertently learn.

3.6.5 Structural overlaps and ambiguity in biological features

In images of fish larvae, anatomical structures can appear blurred, overlap, or lack clear boundaries, making it difficult for models to distinguish them accurately. These ambiguities are especially problematic in tasks like segmentation and landmark detection,

where precise identification of structure boundaries is critical. Overlapping structures or low-contrast regions may lead models to misclassify or overlook important features, affecting overall performance.

3.6.6 Tolerance to data corruption

Bioimage datasets can contain mislabeled or corrupted images due to errors during data collection or annotation. DL models are often sensitive to such corruption, which can negatively impact model performance. In some cases, models may even learn from these errors, leading to poor generalization and reduced reliability. Developing models that can tolerate or correct for corrupted data without extensive manual intervention remains a major challenge in biomedical imaging.

3.6.7 Choice of image analysis methods and protocols

Traditional image analysis methods that rely on basic image processing functions and hand-crafted features, work well for small datasets. However, as the amount of data and its dimensionality increases, these methods become time-consuming and demand significant human intervention and technical expertise for the analysis. In contrast, deep learning methods, can significantly accelerate the analysis process by enabling semi or fully automated workflows. Despite this advantage, training deep-learning-based convolutional neural networks (CNNs) for high-content image analysis remains a complex challenge, and effectively applying these techniques in biomedical image analysis tasks continues to pose difficulties.

3.6.8 Need for a common image analysis platform

In contemporary bioimage analysis, the integration of deep learning models into a common image analysis platform is vital for streamlining workflows and enhancing research outcomes. This platform serves as a centralized environment, where researchers can easily access, visualize, and analyze bioimages using sophisticated algorithms. The proposed image analysis platform should be designed to facilitate the end-to-end processing of bioimages, encompassing stages from image acquisition and preprocessing to model deployment and result interpretation. By providing a user-friendly interface, the platform allows researchers, even those with limited programming expertise, to utilize advanced deep learning techniques without the need for extensive technical knowledge.

3.7 Conclusion

This chapter provides a comprehensive review of common computer vision techniques, encompassing both traditional methods and AI-driven machine learning (ML) and deep learning (DL) algorithms and tools. It emphasizes their applications in biomedical and aquaculture research, particularly in phenotype and morphometric studies related to bone development. Additionally, it addresses the challenges encountered in bioimage analysis, including the complexities of processing high-content and high-throughput imaging data, the limitations of traditional analysis methods, the shortage of well-annotated datasets, and the intricacies involved in implementing effective deep learning architectures

Given the existing challenges in bioimage analysis for morphometric and phenotype studies, this thesis is motivated by the need to develop and implement novel deep learning methodologies that can effectively address these issues. The goal is to create robust, scalable solutions that streamline the analysis process, enhance the accuracy of results, and improve accessibility for researchers across various disciplines. By focusing on the development of automated image analysis methods tailored to aquaculture and biomedical research, this work aims to contribute significantly to the field. The integration of a common image analysis platform will empower researchers to leverage advanced deep learning techniques without extensive technical expertise, promoting collaboration and innovation in the study of biological processes. Ultimately, this research endeavors to provide new insights into morphometric and phenotypic studies, advancing our understanding of fish development and contributing to the broader field of bioinformatics.

Table 3.1: Image Analysis Tools and their specifications.

Name	Ref. paper	Image modality	Fish type and age	Application	Open source?	Data availability	Research area
EmbryoNet	[30] D. Capek et al.	Light microscopy	Zebrafish (1hpf *), Medaka (0-48 hpf), Stickleback (0-140 hpf)	Phynotype classification	Yes	Yes	Biomedical
QuantiFish	[179] David R. Stirling et al.	Fluorescent microscopy	Zebrafish (4dpf **)	Infection dissemination analysis	Yes	On request	Biomedical
ZF-AutoML	[164] R. Sawaki et al.	Light microscopy	Zebrafish larvae (0–96 hpf)	Phenotype classification	Yes	No	Biomedical
ZFTool	[33] M. J. Carreira et al.	Fluorescent green channel microscopy	Zebrafish larvae (0–72 hpf)	Toxicity screening	Yes	No	Biomedical
ZF-Mapper	[219] D. Yamamoto et al.	Light microscopy	Zebrafish larvae (2–6 dpf)	Cancer Xenograft Zebrafish screening	Yes	Yes	Biomedical
ZebraZoom	[131] O. Mirat et al.	Time series (2D+t)	Zebrafish larvae (5–7 dpf)	Locomotion tracking, Behavioral analysis	No	No	Biomedical
FishInspector	[190] E. Teixido et al.	Light microscopy	Zebrafish embryos (0–96 hpf)	Phenotype screening	Yes	Yes	Biomedical
Stytra	[178] V. Stih et al.	Time series (2D+t)	Zebrafish larvae	Zebrafish tracking, Behavioral analysis	Yes	Yes	Biomedical
ZebrafishMiner	[153] M. Reischl et al.	Fluorescent microscopy	Zebrafish embryos and larvae (32 hpf)	Fluorescent quantification in body parts	No	No	Biomedical
CNNTracker	[230] X. Zhiping et al.	Fluorescent microscopy	Zebrafish (over 6 months)	Zebrafish tracking	Yes	No	Biomedical
ZFBone	[188] M. Tarasco et al.	Fluorescent microscopy	Zebrafish larvae (6 dpf)	Morphometric analysis	Yes	Yes	Biomedical
IMAFISH_ML	[137] A. Navarro et al.	Microscopy, RGB, radiography	Adult gilthead seabream, meagre, red porgy	Morphometric analysis	Yes	No	Aquaculture

Continue on the next page

Table 3.1: Image Analysis Tools and their specifications (cont.).

Name	Ref. paper	Image modality	Fish type and age	Application	Open source?	Data availability	Research area
DeepLabCut	[184] Suryanto et al.	Video frames	All types	Landmark prediction	Yes	Yes	Biomedical, Aquaculture
Cytomine	[124] R. Marée et al.	All types	All types	General image analysis	Yes	Yes	Biomedical, Aquaculture

* hours post fertilization

** days post fertilization

CHAPTER 4

SEGMENTATION IN MICROSCOPY BIOIMAGES OF ZEBRAFISH

As we discussed in chapter 1, biomedical research heavily uses Zebrafish (*Danio rerio*) as a model to study developmental processes. In the earlier stage of their lifecycle, zebrafish embryos and larvae are completely transparent, which greatly facilitates monitoring of their developmental organs such as operculum and vertebral column using microscopy techniques [56, 75, 130]. Biomedical researchers also rely on microscopy to study the effects of various chemical compounds on the developing parts of the fish model in toxicological studies [34]. Such analyses often involve segmenting different categories of regions of interest (ROI) within images in order to quantify their morphological changes. For example, the analysis of *Head* and *Operculum* (a series of bone) regions of Zebrafish larvae and the quantification of the operculum-to-head ratio is considered as a good marker of increased bone formation and mineralization and it is a validated method to screen for bioactive compounds which have effects on bones [108, 189]. It also gives an additional information on the possible toxicity of a compound at the organism level. However, the visual examination and area quantification are a bottleneck and prevent applying such a workflow at high throughput. In this chapter, supervised deep learning strategies are proposed and evaluated to segment head and operculum regions, as evaluation of such approaches has not been proposed previously.

Reference: This chapter is an adapted version of the work we published in "Navdeep Kumar, Alessio Carletti, Paulo J Gavaia, Leonor M Cancela, Marc Muller, Pierre Geurts, Raphaël Marée, **"Deep Learning Approaches for Head and Operculum Segmen-**

tation in Zebrafish Microscopy Images", International Conference on Computer Analysis of Images and Patterns (CAIP-2021)".

Demo server with Datasets: https://github.com/navdeepkaushish/S_Zebrafish_Head_Operculum_UNet_Segmentation

4.1 Introduction

In this chapter, we propose variants of deep learning methods to segment head and operculum of the zebrafish larvae in microscopic images. In the first approach, we used a three-class model to jointly segment head and operculum area of zebrafish larvae from background. In the second, two-step, approach, we first trained binary segmentation model to segment head area from the background followed by another binary model to segment the operculum area within cropped head area thereby minimizing the class imbalance problem. Both of our approaches use a modified, simpler, U-Net architecture, and we also evaluate different loss functions to tackle the class imbalance problem. We systematically compare all these variants using various performance metrics.

Our methodology is detailed in Section 4.2, beginning with an overview of the image acquisition settings and datasets in Section 4.2.1. The proposed deep learning strategies and CNN architecture are then discussed in Sections 4.2.2 and 4.2.3, respectively. The evaluation protocol for this work is outlined in Section 4.2.4, followed by the presentation of results in Section 4.3.

4.2 Methodology

This section describes the image acquisition procedure and dataset, followed by two deep learning strategies. Additionally, it provides details on the convolutional neural network (CNN) architectures used to segment the head and operculum regions and the experimental protocol we followed for this work.

4.2.1 Image Acquisition and Dataset Description

Zebrafish larvae stained with alizarin red S were imaged using a MZ 7.5 fluorescence stereomicroscope (Leica, Wetzlar, Germany) equipped with a green light filter (λex=530–560 nm and λem=580 nm) and a black-and-white F-View II camera (Olympus, Hamburg, Germany). Images were acquired using the following parameters: exposure

time 1s, gamma 1.00, image format (1376 × 1032) pixels, binning (1 × 1). For morphometric analysis, color channels of the RGB images were split. and red channel (8-bit) images were used for further analyses.

We follow a supervised deep learning approach that requires original images and corresponding head and operculum ground-truth masks to design and validate the approach. Our dataset consists of 8-bit single channel (red channel) fluorescence images of zebrafish larvea at 6 days post fertilization (dpf). Red channel fluorescence images were first transformed into greyscale images (with contrast and brightness enhancement) to ease the manual annotations by experts of head and operculum areas. Manual annotations (illustrated in Figure 4.1) consist of green and red contours of head area and operculum area respectively. A total of 2293 zebrafish images of 1376×1032 resolution have been collected and manually annotated over a period of one year. The dataset consists of 28 different sets of experiments using 5 different compounds, to analyse their effect on the operculum of the zebrafish larvea. Each set has been acquired with the same acquisition settings. Manual annotations are then imported into Cytomine open-source software [124] to centralize data and ease binary masks creation to be further used as inputs of deep learning algorithms. In supervised learning setting, a segmentation method

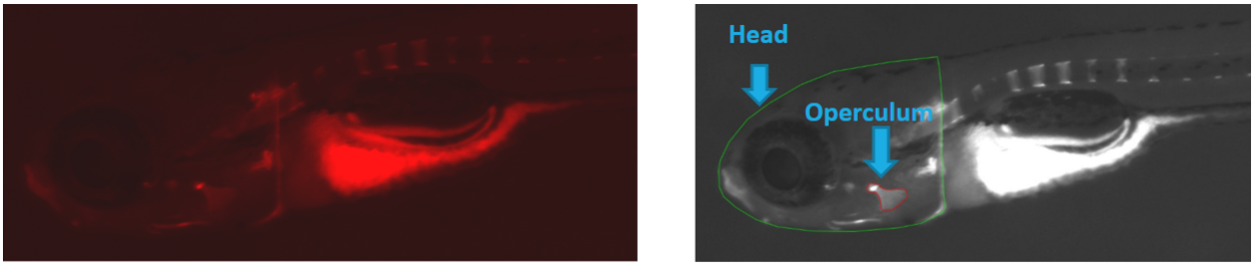

Figure 4.1: Image sample and its corresponding head and operculum annotations. (a) Raw red channel image (b) manually annotated gray-scale version of the sample image.

requires segmentation masks as its label for learning. Therefore, it is necessary to create the head and the operculum masks from their respective contours. Firstly, contours of the head and operculum regions are extracted using OpenCV's image processing function *'findContours'*, after that segmentation masks are created using "flood_fill" algorithm [138] to fill the contour area with white pixels (255 value) and rest of the area is treated as background and filled with black pixels (0 value). At the end of this procedure, we get binary segmentation masks for head and operculum as shown in Figure 4.2.

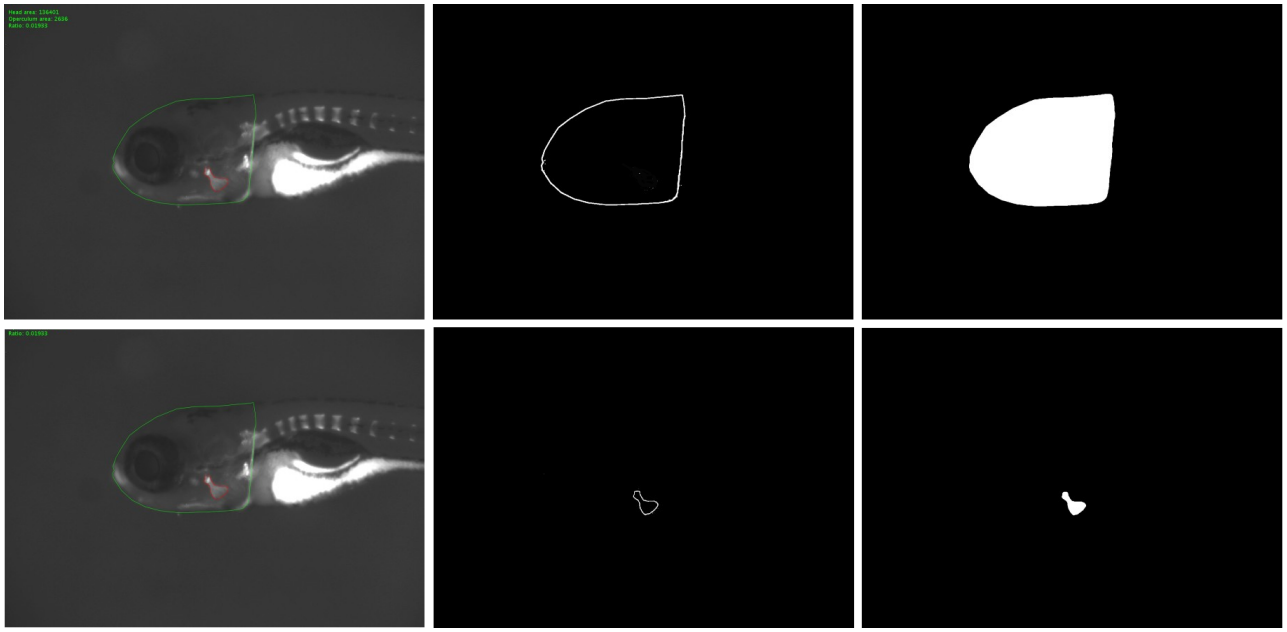

Figure 4.2: Images and ground truth masks. Left: Two greyscale images. Middle: contour extraction of head and operculum areas. Right: binarized ground truth masks.

4.2.2 Two deep learning strategies

In this section, we explore two deep learning strategies, we implemented in this work. The first strategy called "One-Step segmentation with a three class model" is design to segment all the three classes namely head, operculum and background in one go. The second strategy called "Two-step segmentation with two binary models" is designed to firstly predict head with a binary class model and then operculum with another second binary class model. The detailed description of the two strategies is discussed in the next two sections.

4.2.2.1 One-step segmentation with a three-class model.

Following this strategy, original size images without cropping are used. Since typical CNN networks require input images of small size (see below), original sized images are first downsized to the size required by the network, keeping their original aspect ratio to avoid any kind of distortions in the predictions, while upsizing the predicted masks. Since our images are rectangular but network require square images, we padded the rectangular images with zeros to make them square. A three-class output segmentation model is then trained to segment both head and operculum from background areas as illustrated in Fig. 4.3 (top).

4.2.2.2 Two-step segmentation with two binary models.

In this approach, a first binary segmentation model is trained to segment the head from the background in original full images downsized appropriately (as in the three-class approach). A second binary segmentation model is trained to segment the operculum area using resized cropped images (rectangular box around the head). At prediction phase, the first model is applied to segment the head, then a rectangular bounding box is automatically extracted. Using these box coordinates, we apply the second model to the resized cropped images (around the head) to segment the operculum area. The two-step approach is illustrated in Fig. 4.3 (bottom).

4.2.3 U-Net Implementation

For both approaches, the U-Net architecture [157], we discussed in section 2.4.4.3 has been adapted to segment areas of the zebrafish larvea. As a reminder, the main idea of U-Net is its two parts: the convolution (encoder) or contracting operations, and deconvolutional (decoder) or expanding operations. In the first part, convolutional operations are applied in successive layers with the max pooling operations at the end of each layer, thereby contracting the input resolution. In the second part, an expanding resolution path is adopted using upsampling or deconvolutional layers. The first part is considered as a traditional stack of convolutional and max pooling layers to capture context information within the image. In the second part, deconvolutional operations are applied along a symmetric expanding path to capture the precise localized information. One more important thing about this architecture is its symmetric concatenation of the previous activations from the first part to the activations of the second part.

As preliminary results with the original U-Net architecture on the training set were unsatisfactory (including a tendency to predict only the majority class, i.e. the background), we implemented some modifications in U-Net architecture including the input size and output size of the network and number of layers and filters. Fig. 4.4 shows our "modified U-Net" network architecture. In our experiments, we used two versions of modified U-Net, one that accepts 512×512 images as input and another that accepts 256×256 images. In both cases, the output size of the masks is the same as the input size whereas in [157], authors used 572×572 inputs and 388×388 outputs. The reason behind using two variants of the network is to assess whether using less parameters will negatively impact recognition performance. Using smaller networks indeed reduces execution times which can be useful in real-time applications. With the

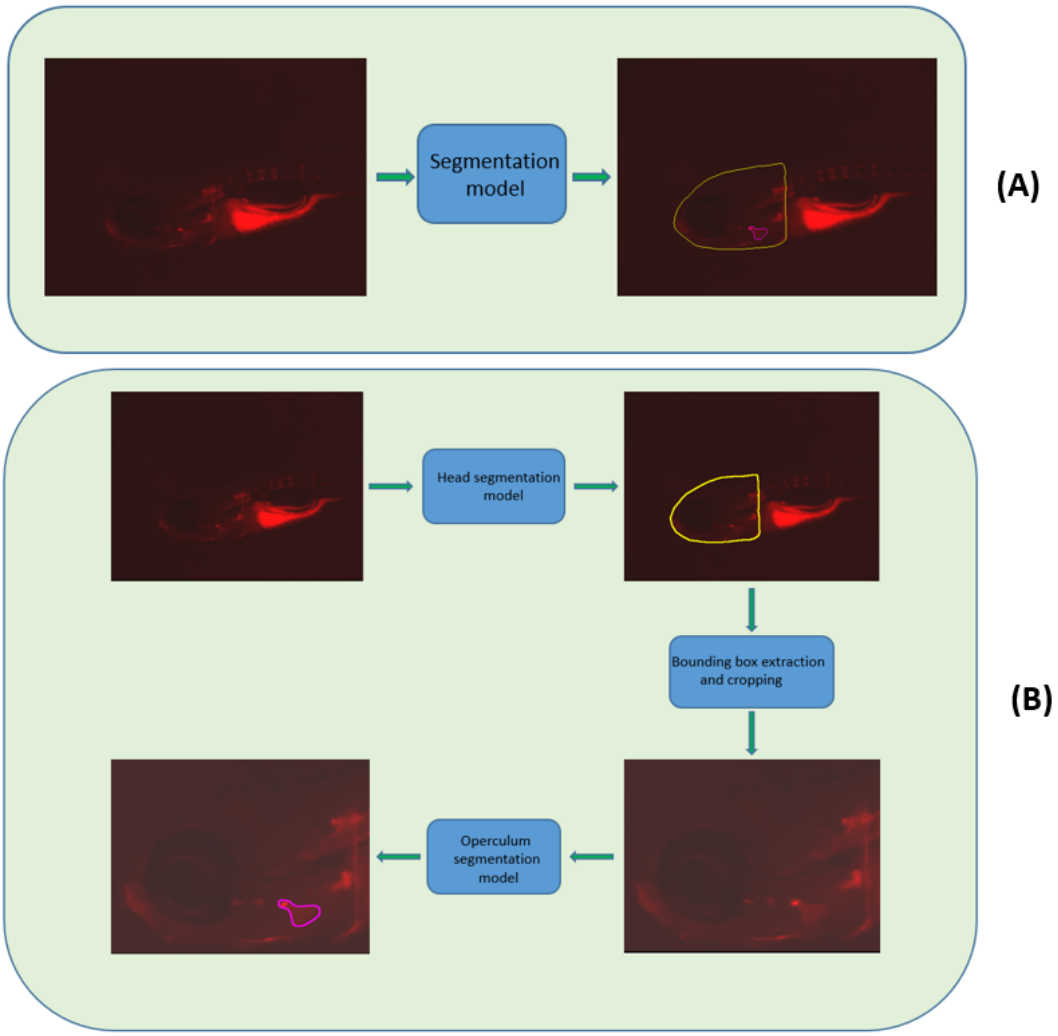

Figure 4.3: (A) One-step segmentation approach with three classes: head (yellow contour), operculum (pink contour), background. (B) Two-step binary segmentation approach with a first binary model (head vs background) followed by a second binary model (operculum vs other).

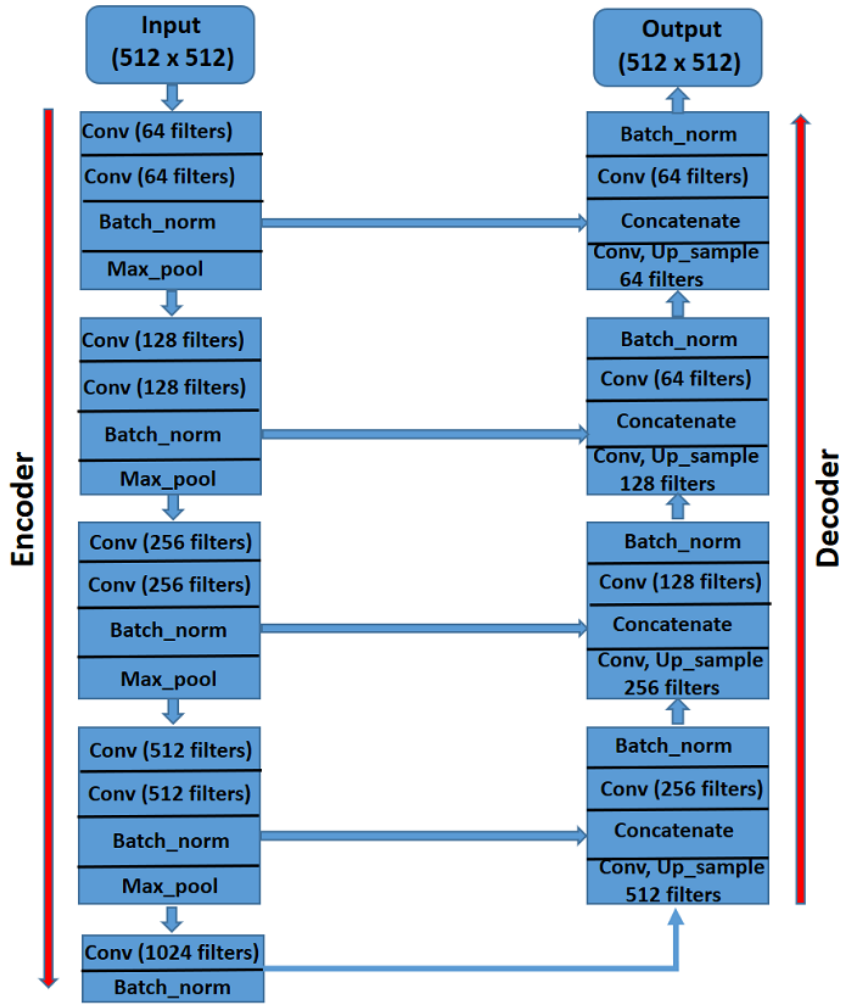

Figure 4.4: Modified U-Net architecture used in experiments for 512×512 sized images.

small size variant of the U-Net architecture (with 256×256 input size), we used fewer filters in each convolutional block as compared to the larger network thereby reducing the network size and the number of parameters by 5 folds. For better optimization, we used "Adam" [98] optimizer and batch normalization in each convolutional block before max pooling. Adam uses "gradient descent with momentum" combined with an adaptive learning rate using exponential moving averages, which makes it more computationally and memory efficient than "Stochastic Gradient Descent" used in the original U-Net paper. During training, we also used data augmentation (random flips and rotations, brightness, and contrast changes). We implemented these networks in Python using Tensorflow and Keras [42].

4.2.4 Experimental protocol

We first split the dataset into 2105 images for training and 188 images for final evaluation. To assess variability, the set of 2105 images is split into five equally sized folds. Each fold is used in turn as the validation set and the remaining folds as the training set. Five models are trained independently on each training set for 1000 epochs and the five best models on their corresponding validation set across the epochs are finally retained as the final models. In addition, we used early stopping, which forces the training to stop when there is no improvement in the training loss for 100 consecutive epochs. Another callback called "model checkpointing" is also used in which current training model is saved if it is better than the previous one on the validation set. We report in tables, the average performance and standard deviation of these five models estimated on the test set with 256 × 256 (see tables 4.2 and 4.1) and 512 × 512 (see tables 4.4 and 4.3) input image resolution using three class and two step binary class settings.

In both approaches, we used deep learning based semantic segmentation approach in which a model predicts the class of every pixel in the image (dense predictions). In such a setting, we are faced with a problem of class imbalance as less than 2% of the image area is occupied by operculum region while around 90% is background region. In such situation, the contribution of the majority class (in our case, the background) in the loss during training is more important than that of the minority class, which biases the model in favor of the majority class while ignoring minority class. While the two-step approach tends to reduce this phenomenon (by cropping then predicting operculum only within the head region), a certain class imbalance still persists. Therefore, for both approaches we propose to evaluate different loss functions during training to handle class imbalance. Namely, we evaluated the Cross Entropy Loss, Dice Loss, Tversky Loss, Focal Loss and Jaccard Loss [21]. We have discussed about these loss functions in Section 2.6.2.

4.3 Results and Discussion

Tables 4.1 and 4.3 show the results of the first (*three-class* segmentation) approach whereas tables 4.2 and 4.4 of the second (*two-step binary class segmentation*) approach using 256×256 input size and 512×512 input size networks, respectively. In both variants, we report several performance metrics that take into account class imbalance, namely *Precision, Recall*, *F1 Score* and *Dice score*, computed at the pixel level and averaged over the 5 models. To get a single score with which to compare the models, the Dice score is

further averaged over head and operculum. Its standard deviation over the 5 models is also provided to assess variability.

Table 4.1: Segmentation results with the one-step, three-class approach using different loss functions for input size 256×256.

Avg. scores with three-class output based segmentation					
Loss function	Class	Precision	Recall	F1 Score	Dice Score±S.D.
Cross Entropy	Head	0.9806	0.9796	0.9801	0.9412±0.0043
	Operculum	0.8780	0.9263	0.9014	
Tversky loss	Head	0.9779	0.9806	0.9792	0.9470±0.0017
	Operculum	0.9086	0.9190	0.9136	
Dice loss	Head	0.9819	0.9806	0.9813	0.9462±0.0024
	Operculum	0.9120	0.9092	0.9106	
Jaccard Loss	Head	0.9678	0.9789	0.9733	0.49±0.0002
	Operculum	0.0	0.0	0.0	
Focal loss	Head	0.9820	0.9798	0.9809	0.9442±0.0046
	Operculum	0.9060	0.9076	0.9066	

Table 4.2: Segmentation results with the two-step, binary approach using different loss functions for input size 256×256.

Avg. scores with two binary-class output based segmentation					
Loss function	Class	Precision	Recall	F1 Score	Dice Score±S.D.
Cross Entropy	Head	0.9832	0.9805	0.9819	0.9540±0.0015
	Operculum	0.9196	0.9340	0.9267	
Tversky loss	Head	0.9824	0.9806	0.9815	0.9524±0.0024
	Operculum	0.9104	0.9374	0.9237	
Dice loss	Head	0.9828	0.9826	0.9827	0.9511±0.0046
	Operculum	0.9175	0.9276	0.9225	
Jaccard Loss	Head	0.9782	0.9826	0.9804	0.9513±0.0012
	Operculum	0.9124	0.9355	0.9238	
Focal loss	Head	0.9835	0.9815	0.9825	0.9516±0.0018
	Operculum	0.9213	0.9261	0.9236	

In the three-class approach, the tversky loss seems to better cope with the strong class imbalance in both 512×512 and 256×256 settings. The worst performer in the three-class approach is Jaccard loss as it only predicts the majority class (90% background) and no operculum area. This loss leads however to good predictions with the two-step binary approach in both input size settings. In the two-step binary segmentation approach, all losses are very close except cross entropy in 512×512 setting. Overall, the two-step approach for 512×512 inputs with Jaccard loss has a slight edge over other losses. We

Table 4.3: Segmentation results with the one-step, three-class approach using different loss functions for input size 512×512.

Avg. scores with three-class output based segmentation					
Loss function	Class	Precision	Recall	F1 Score	Dice Score±S.D.
Cross Entropy	Head	0.9815	0.9747	0.9781	0.9358±0.0064
	Operculum	0.8992	0.8934	0.8953	
Tversky loss	Head	0.9812	0.9789	0.9800	0.95±0.0011
	Operculum	0.9090	0.9308	0.9196	
Dice loss	Head	0.9822	0.9744	0.9783	0.9428±0.0043
	Operculum	0.9085	0.9066	0.9074	
Jaccard Loss	Head	0.9678	0.9789	0.9733	0.49±0.0002
	Operculum	0.0	0.0	0.0	
Focal loss	Head	0.9817	0.9768	0.9792	0.9364±0.007
	Operculum	0.9078	0.8846	0.8946	

Table 4.4: Segmentation results with the two-step, binary approach, using different loss functions for input size 512×512.

Avg. scores with two binary-class output based segmentation					
Loss function	Class	Precision	Recall	F1 Score	Dice Score±Std.
Cross Entropy	Head	0.9840	0.9780	0.9810	0.9189±0.0159
	Operculum	0.9114	0.8428	0.8747	
Tversky loss	Head	0.9832	0.9785	0.9808	0.9505±0.0024
	Operculum	0.9223	0.9245	0.9234	
Dice loss	Head	0.9828	0.9797	0.9812	0.9424±0.0057
	Operculum	0.9256	0.8947	0.9097	
Jaccard Loss	Head	0.9818	0.99796	0.9807	0.9516±0.002
	Operculum	0.9178	0.9311	0.9244	
Focal loss	Head	0.9841	0.9732	0.9786	0.9490±0.0031
	Operculum	0.9207	0.9227	0.9217	

believe that the improved performance of the two-step approach is due to the fact that the second segmentation model works with a cropped, head-focused, dataset. Because of the cropping, the class imbalance is not as severe and the operculum image is not downscaled as much as with the three-class approach. Predictions are thus more precise and less influenced by the class imbalance. Regarding the two input sizes, we see that they lead to almost identical performance in terms of Dice Score. Sample predictions from the best performing models are shown in Figure 4.5. To further evaluate approaches with respect to their actual intended use, Table 4.5 compares the ground truth and predicted operculum-to-head ratios using the best performing models from both approaches. We used four metrics: mean squared error, Pearson and Spearman correlations, and a fourth

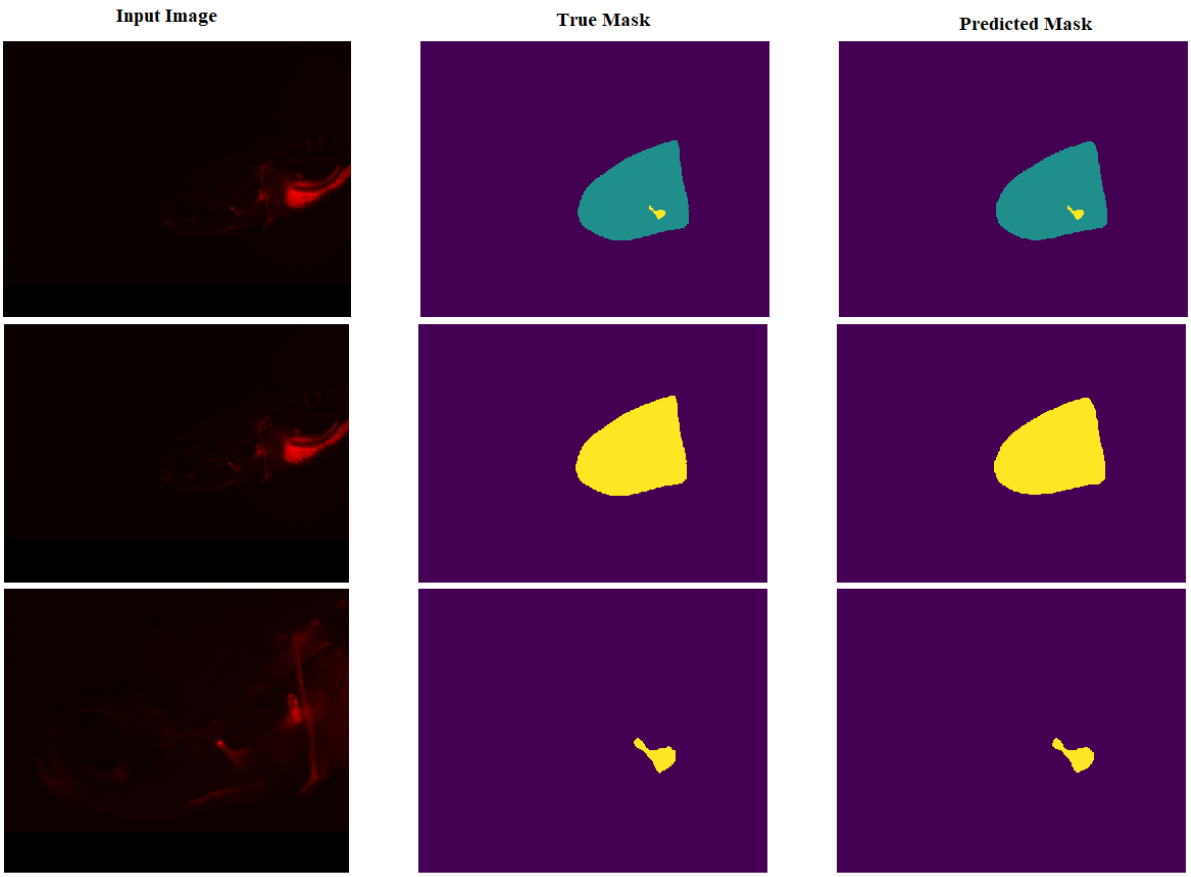

Figure 4.5: Sample predictions with best performer on test images with three class (top row) and two-step binary class (last two rows). From the first to third column: input Image, true mask, predicted mask.

custom metric P called proximity measure. We compute a proximity measure to see how close our predicted ratios to the actual ratios for test images. Proximity measure is calculated as first measuring fraction $\frac{Min(True\_ratio,Predicted\_ratio)}{Max(True\_ratio,Predicted\_ratio))}$ over each test image and then taking the mean over all the test images. If that number is closes to 1 it means that our predicted ratio is very close to actual one and if it goes far away then it signifies the predictions are bad. One can see that the three-class approach performs better in terms of Pearson correlation and MSE, while the two-step approach performs better in terms of Spearman correlation. In terms of proximity measure P, both the approaches are close to 1 which signifies that our model predictions are quite satisfactory. While the two-step approach works better at the pixel level, this result suggests that further validation is required to assess which one of the two methods will be the most appropriate in the context of actual morphological studies.

Table 4.5: Evaluation of operculum-to-head ratio predictions using best performing models.

	MSE	Pearson	Spearman	P
Three-class approach	$1.014e-5$	0.210	0.270	0.8577
Two-step approach	$4.228e-4$	0.117	0.314	0.8468

4.3.1 Robustness to image acquisition with another microscope

In practice, microscopes with different acquisition settings might be used over time by biomedical researchers, which raises the issue of robustness of segmentation models to such variabilities, an issue known as domain shift [70] (also see Section 2.5). As a first step towards robustness evaluation, we applied our best two-step binary approach on additional, unlabeled, images acquired with another microscope namely Leica MZ10F fluorescence stereomicroscope equipped with a green fluorescence filter (λex= 546/10 nm), a barrier filter (λem =590 nm) and a DFC7000T camera (Leica, Wetzlar, Germany) with a different output image size (1920×1440). When run on these unprocessed new images, we observe that the performance of our model declines, as illustrated by Fig. 4.6 (first row).

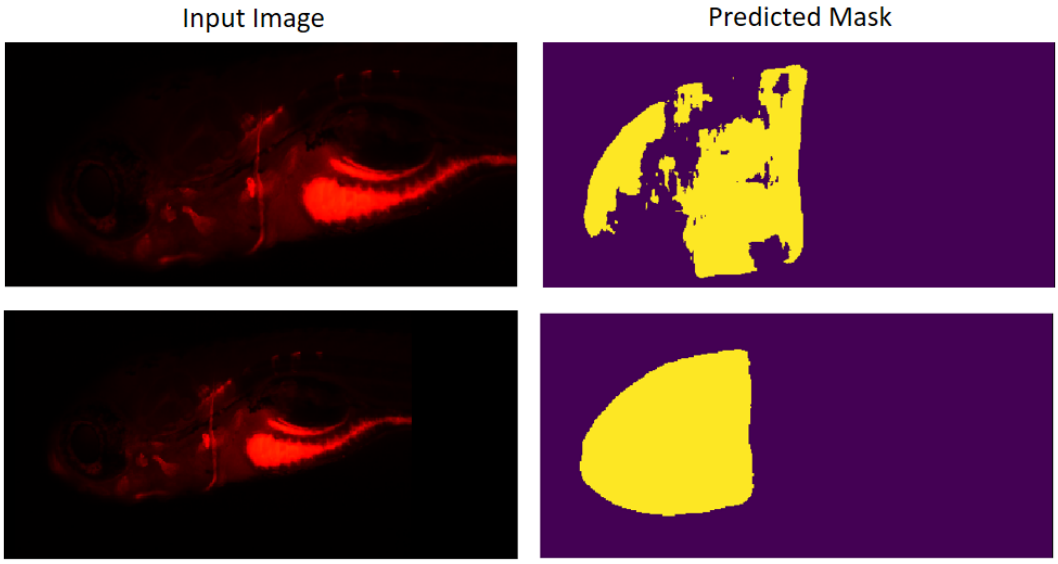

Figure 4.6: Robustness evaluation: Predictions from best model using two-step binary class approach on a new image acquired with another microscope before pre-processing (first row), and after pre-processing (second row).

We hypothesized that this is due to the fact that, in the new microscope setting,

ROIs (fish head and operculum) are larger in proportion to the size of the full image as compared to ROIs in the original training images. To address this issue, we applied a very simple *pre-processing* step to reduce the scale proportion of ROIs in the image. First, we downscaled the new images to the resolution of the original images (i.e., from 1920×1440 down to 1376×1032) keeping the same aspect ratio. We then centered the resulting 1376×1032 image into a 1920×1440 image, filling the new pixels with zeros. Figure 4 (second row) illustrates the positive effect of this pre-processing on the prediction. Note that downscaling further the image in the first step does not seem to affect the performance. We hypothesized that this is due to the use of pooling layers that makes network features somewhat scale invariant (in the direction of a decrease of resolution at least). In practice, this scale calibration step would require a human expert to manually draw a rectangle around the head within a single image when a new microscope is used to initiate the automatic rescaling for the whole set of new images (so that the bounding box is rescaled down to the average size of the head in the learning set images). We consider this manual intervention to be acceptable.

4.4 Conclusion

We have evaluated deep learning based semantic segmentation variants on a new dataset of more than two thousands fluorescent microscopy images of Zebrafish larvae where the goal is to quantify operculum-to-head ratio. The dataset and prediction code compatible with Cytomine open-source web platform [124] is available to foster further research and to enable biomedical experts to routinely use our developments and proofread predictions. We plan to use such developments as the basis of large-scale morphological studies where the effects of different concentrations of many compounds on bone formation and mineralization will be evaluated thoroughly using various statistics (such as operculum-to-head ratio) derived from predicted masks. In the future, it may be necessary to investigate more advanced approaches for other image variations due to change of acquisition setting but ours was sufficient on the new microscope used by our collaborators.

CHAPTER 5

ANATOMICAL LANDMARK DETECTION IN FISH BIOIMAGES

In numerous bioimage studies, identifying anatomical landmarks is an essential step for conducting morphometric analyses and measuring the shape, volume, and size characteristics of the organism being examined [84]. Landmarks are geometric keypoints localized on an "object" and can be described as coordinate points in a 2D or a 3D space. For example, in human cephalometric study, human cranium is analyzed for diagnosis and treatment of dental disharmonies [133] using X-Ray medical imaging techniques. In biomedical research where fish species such as Zebrafish *(Danio rerio)* and Medaka *(Oryzias letipes)* are used as models, various morphometric analyses are performed to quantify deformities in them and further identify cause and treatment for human related bone disorders [91, 215]. Such studies require to analyze and classify deformities in the vertebral column, jaws or caudal fin of the fish, which is addressed by first detecting specific landmark positions in fish images. In aquaculture industry, food fish such as gilthead Seabream suffer from bone related disorders due to the non-natural environment in which they are reared and morphometric studies are carried out to quantify these deformities [58, 205]. Such studies also require the researchers to select and mark some important landmark locations on fish images in order to perform external shape analyses [118]. In this chapter, we perform an empirical evaluation of variants of deep learning methods to automatically localize anatomical landmarks in bioimages of fishes acquired using different imaging modalities (microscopy and radiography). To our knowledge,

our work is one of the first few attempts to implement a fully automatic end-to-end deep learning based method for the task of landmark detection in heterogeneous fish bioimages.

Reference: This chapter is adapted from our publication "Navdeep Kumar, Zachary Dellacqua, Claudia Di Biagio, Ratish Raman, Arianna Martini, Clara Boglione, Marc Muller, Pierre Geurts, Raphaël Marée, **"Empirical Evaluation of Deep Learning Approaches for Landmark Detection in Fish Bio-Images"**, European Conference on Computer Vision Workshops (ECCV- 2022)."

Demo server with datasets: http://research.uliege.cytomine.org/ (username: eccv2022bic password: deep-fish)

5.1 Introduction

Manual annotations of landmarks locations are very labour intensive and require dedicated human expertise. The emergence and heterogeneity of high-throughput image acquisition instruments makes it difficult to continue analyzing these images manually. To address the problem, biomedical researchers began to use automatic landmark localization techniques to speed up the process and analyze large volumes of data. Conventional landmark detection techniques use image processing in order to align two image templates for landmark configurations then applying some Procrustes analysis [24]. Classical machine learning techniques such as random forest based algorithms were also proposed in [177] [112] [202] to automatically localize landmarks in microscopy images of zebrafish larvae.

Recently, landmark detection or localization has also been extensively studied in the broader computer vision field, especially for real time face recognition systems [94][218][71], hand-gesture recognition [151], and human pose estimation [162][12]. With the advent of more sophisticated techniques such as deep-learning based Convolutional Neural Networks (CNNs), the performance of computerized models for object detection and classification has become comparable to human performance. While deep learning models reach a high level of accuracy in computer vision tasks with natural images (e.g. on ImageNet), there is no guarantee that these methods will give acceptable performance in specific bioimaging applications where the amount of training data is limited. Indeed, learning landmark detection models requires images annotated with precise landmark positions while experts to carry out these annotations are few, the annotation task is tedious and it must be repeated for every new imaging modality and biological entity.

In this chapter, we want to evaluate state-of-the-art deep learning based landmark detection techniques to assess if they can simplify and speed up landmark analyses in real-world bioimaging applications, and to derive guidelines for future use. More precisely, we evaluate the two main families of methods in this domain, namely direct multivariate regression and heatmap regression, and we focus our experiments on the identification of anatomical landmarks in 2D images of various fish species. Section 5.2 reviews prior research on anatomical landmark detection in bioimages. In Section 5.3, we outline our datasets and image acquisition settings. Our methodologies, network architectures, and evaluation protocol are detailed in Section 5.4. Finally, we present and discuss the empirical results in Section 5.5.

5.2 Related Work

In biomedical image analysis, patch-based deep learning methods are proposed in which local image patches are extracted from the images and fed to the CNN to detect the landmark locations [14, 172]. Patch-based methods are usually used to train one landmark model for each landmark location making the whole process computationally very expensive. These models often require plenty of memory storage to operate if the number of landmark points to detect is high. Another drawback of using the patch-based methods is missing global information about all the landmarks combined as local patches represent only limited contextual information about the particular landmark.

Among end-to-end deep learning approaches, the first prominent solution is to output directly the (x, y) coordinates of the landmarks using CNNs regressors [105]. These direct coordinate regression based methods are very simple to design and faster to train. However, to get optimal performances, this approach generally requires large training datasets and deeper networks [81]. Another approach is to output heatmaps corresponding to the landmark locations [55, 143, 144]. In this scenario, heatmaps are generated from the labeled landmarks locations during training and CNNs are trained to predict these heatmaps. These heatmaps encode per pixel confidence scores for landmark locations rather than numbers or values corresponding to landmark coordinates. The most common heatmap generation methods employ distance (linear) functions or some non-linear gaussian or exponential kernels [223]. In [81] and [121], the authors proposed a method that combines the heatmap based regressors with direct coordinate regressors to automatically localize landmarks in MRI images of spine.

The data scarcity in biomedical image analysis is one of the biggest concerns as it is

difficult to train a deep CNN from scratch with limited amount of images and ground truths. To address this issue, the authors of [142, 172] explore transfer learning methods such as using a pre-trained CNN as backbone and only training or fine-tuning its last layers for the problem of cephalometric landmark detection. Transfer learning is also used in animal behaviour studies in neuroscience where landmarks are used to aid computer-based tracking systems. [127] devised a transfer learning based landmark detection algorithm that uses pre-trained Resnet50 as backbone to automatically track the movements in video recordings of the animals. To tackle the problem of limited data, the authors of [154] proposed a method to train models on thousands of synthetically generated images from other computer vision tasks such as hand recognition systems and evaluate them on MR and CT images.

There are cases in which two landmark points are either very close to each other or one is occluding another landmark. In these cases, a single CNN model is not sufficient to achieve optimal performance in locating the landmarks. To handle these scenarios, authors in [104, 196] proposed a combination of CNN regressor and Recurrent Neural Network (RNN) in which RNNs are employed to remember the information for landmark locations to further refine the predictions given by the CNN regressor. Although these methods can lead to very good performance for landmark detection, they are very hard to train on limited image data due to their complex architectural design.

5.3 Dataset Description

In this work, we use three datasets acquired using different microscopy and radiography imaging protocols. These datasets contain images of three different fish species, namely zebrafish (*Danio rerio*) and medaka (*Oryzias latipes*), used in biomedical research as model fishes, and gilthead Seabream (*Sparus aurata*), used for aquaculture research. The Zebrafish microscopy dataset is acquired from GIGA Institute at the University of Liège whereas the Medaka microscopy and gilthead Seabream radiograph datasets are acquired from the department of Biology, University of Rome, Tor Vergata. The summary of each dataset and detailed dataset descriptions are given in Table 5.1 .

5.3.1 Zebrafish Microscopy Dataset

This dataset is composed of 113 microscropy images of zebrafish (Danio rerio) larvae at 10dpf (3mm length). Images were captured using an Olympus SZX10 stereo dissecting

Table 5.1: Summary of the datasets used in our methodology

Fish species	Number of images	Number of landmarks	Image modality	Research area
Zebrafish	113	25	Microscopy	Bio-medical Science
Medaka	470	6	Microscopy	Bio-medical Science
Gilthead Seabream	847	19	Radiograph	Aquaculture

microscope coupled with an Olympus XC50 camera with a direct light illumination on a white background. The Olympus XC50 camera allows to acquire 2576 × 1932 pixel resolution images. 25 landmarks are manually annotated by the experts around the head of the zebrafish larvae as folows: 1 and 24: Maxilla; 2 and 23: Branchiostegal ray 2; 3 and 11: Opercle; 4,12,13 and 14: Cleithrum; 5 and 19: Anguloarticular; 6 and 25: Ceratobranchial; 7 and 8: Hyomandibular; 9 and 20: Entopterygoid; 10:Notochord; 21,15 and 18: Parasphenoid; 17 and 22: Dentary; 16: showing anterior end marking. A sample image and its annotations are shown in Figure 5.1.

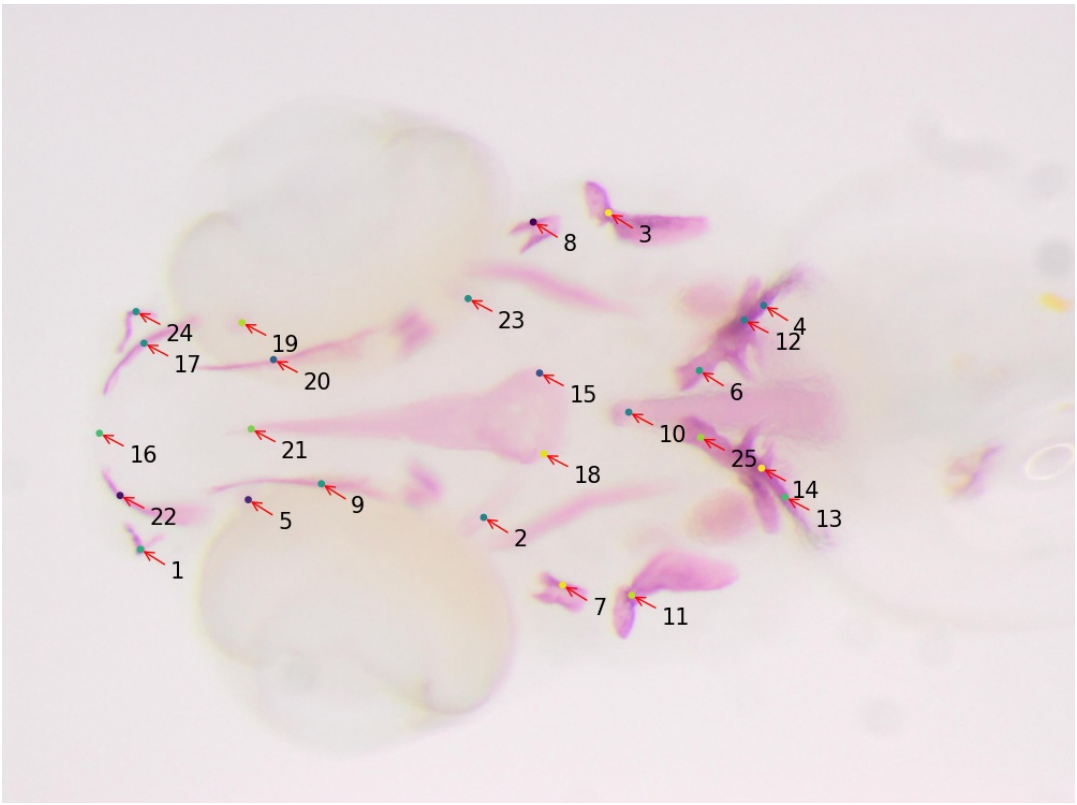

Figure 5.1: Zebrafish image and its annotations from Zebrafish Microscopy Dataset

5.3.2 Medaka Microscopy Dataset

This dataset has 470 images of medaka juveniles (40 days after hatching) where each image has size 2560 × 1920. Samples were *in toto* stained with Alizarin red and photographed with the Camera Axiocam 305 color connected to the AxioZoom V.16 (Zeiss) stereomicroscope. A total number of 6 landmarks are manually annotated as follows: 1: rostral tip of the premaxilla (if the head is bent, the landmark was located between the left and right premaxilla); 2: base of the neural arch of the 1st (anteriormost) abdominal vertebra bearing a rib; 3: base of the neural post-zygapophyses of the first hemal vertebra (*viz.*, vertebra with hemal arch closed by a hemaspine); 4: base of the neural post-zygapophyses of the first preural vertebra; 5: base of the neural post-zygapophyses of the preural-2 vertebra; 6: posteriormost (caudad) ventral extremity of the hypural 1. Figure 5.2 shows a sample image from the medaka dataset with annotated landmarks.

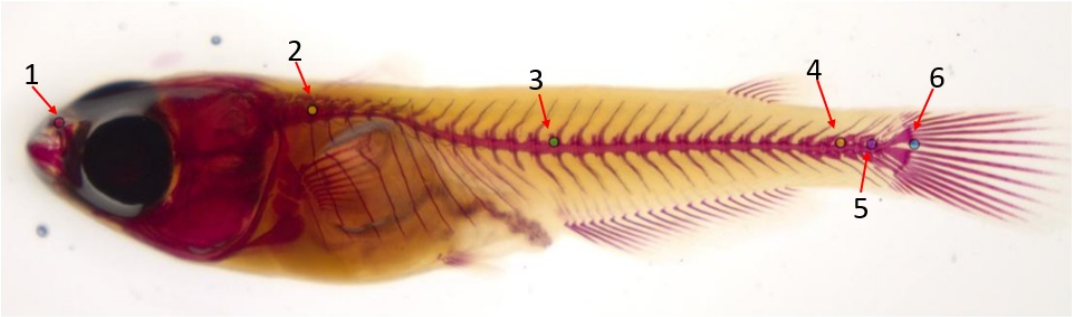

Figure 5.2: Sample image of medaka and its annotations from Medaka Microscopy Dataset

5.3.3 Seabream Radiography Dataset

In this dataset, the fish species is gilthead Seabream (*Sparus aurata*), sampled at 55 gr (average weight). A total of 847 fish were xrayed with a digital DXS Pro X-ray (Bruker) and 19 landmarks are manually annotated on variable image sizes, as follows: A: frontal tip of premaxillary; B: rostral head point in line with the eye center; C: dorsal head point in line with the eye center; D: dorsal extremity of the 1st predorsal bone; E: edge between the dorsal 1st hard ray pterygophore and hard ray; F: edge between the dorsal 1st soft ray pterygophore and soft ray; G: edge between the dorsal last soft ray pterygophore and soft ray; H: dorsal concave inflexion-point of caudal peduncle; I: middle point between the bases of hypurals 2 and 3 (fork); L: ventral concave inflexion-point of caudal peduncle; M: edge between the anal last pterygophore and ray; N: edge between the anal 1st ray

pterygophore and ray; O: insertion of the pelvic fin on the body profile; P: preopercle ventral insertion on body profile; Q: frontal tip of dentary; R: neural arch insertion on the 1st abdominal vertebral body; S: neural arch insertion on the 1st hemal vertebral body; T: neural arch insertion on the 6th hemal vertebral body; U: between the pre- and post-zygapophyses of the 1st and 2nd caudal vertebral bodies. Sample images from the dataset with annotated landmarks are shown in Figure 5.3.

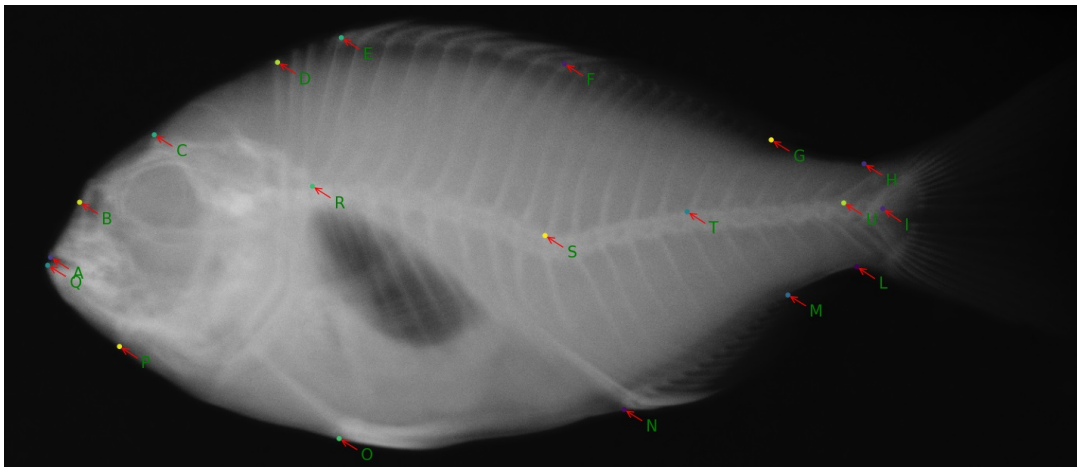

Figure 5.3: Seabream image and its annotations from Seabream Radiography Dataset

5.4 Method Description

We evaluate two deep learning-based regression approaches: direct coordinate regression and heatmap-based regression, which are discussed in detail in Sections 5.4.1 and 5.4.2, respectively. A comprehensive description of the training and prediction phases is provided in Section 5.4.3. Section 5.4.4 covers the CNN architectures used in our methodology, and in Section 5.4.5, we outline the implementation of the experimental protocol.

5.4.1 Direct coordinates regression

Direct coordinate regression is a technique commonly used in anatomical landmark detection, where the model is trained to predict the precise coordinates (typically in the x, y, and possibly z dimensions) of each landmark in an image. Unlike heatmap-based approaches (see Section 5.4.2) that predict the likelihood of each pixel being a landmark and then derive coordinates based on the most probable region, direct

coordinate regression bypasses this intermediate step. Instead, it directly outputs the coordinates of each landmark, streamlining the detection process. In the direct regression approach, the output is designed to predict $(N \times 2)$ numbers, where the first (resp. last) N numbers correspond to x (resp. y) coordinates of the landmarks.

5.4.2 Heatmap-based regression

The second approach is based on outputting the heatmaps (one per landmark) instead of directly predicting the coordinate points for landmark locations. Each heatmap gives information about the likelihood for each pixel of being the location of a particular landmark. At training, the heatmap is constructed to associate to every pixel a score that takes its highest value (1) at the exact location of the landmark and vanishes towards 0 when moving away from the landmark. The size of the region of influence of a landmark is controlled by a user-defined dispersion parameter σ. More formally, and following [223], we have implemented and compared two probability functions to generate these heatmaps, namely a **Gaussian function** F_G and an **Exponential function** F_E, defined respectively as follows:

$$F_G(x,y) = A \cdot \exp\left(-\frac{1}{2\sigma^2}\left((x-\mu_x)^2 + (y-\mu_y)^2\right)\right),$$
$$F_E(x,y) = A \cdot \exp\left(-\frac{\log(2)}{2\sigma}(|x-\mu_x| + |y-\mu_y|)\right),$$

where x and y are the coordinates of a pixel in the image, μ_x and μ_y are the coordinates of the landmark under consideration, σ is the spread of the distribution, and A is a normalizing constant that gives the amplitude or peak of the curve.

To fix the highest score value as 1 at the exact location of the landmark, we set the normalizing constant A to 1, since it corresponds to the maximum value of the gaussian and exponential functions. Figure 5.4 shows the original landmarks on the image (first column) and their corresponding heatmaps, as the superposition of the heatmaps corresponding to each landmark (second and third columns).

5.4.3 Training and prediction phases

In the **training phase**, original images are first downscaled to 256×256 to be fed into the network. Since the original images are rectangular, we first downscale the image to a size of 256 along the largest dimension while keeping the aspect ratio unchanged. Padding is then added to the smallest dimension to produce a 256×256 square image.

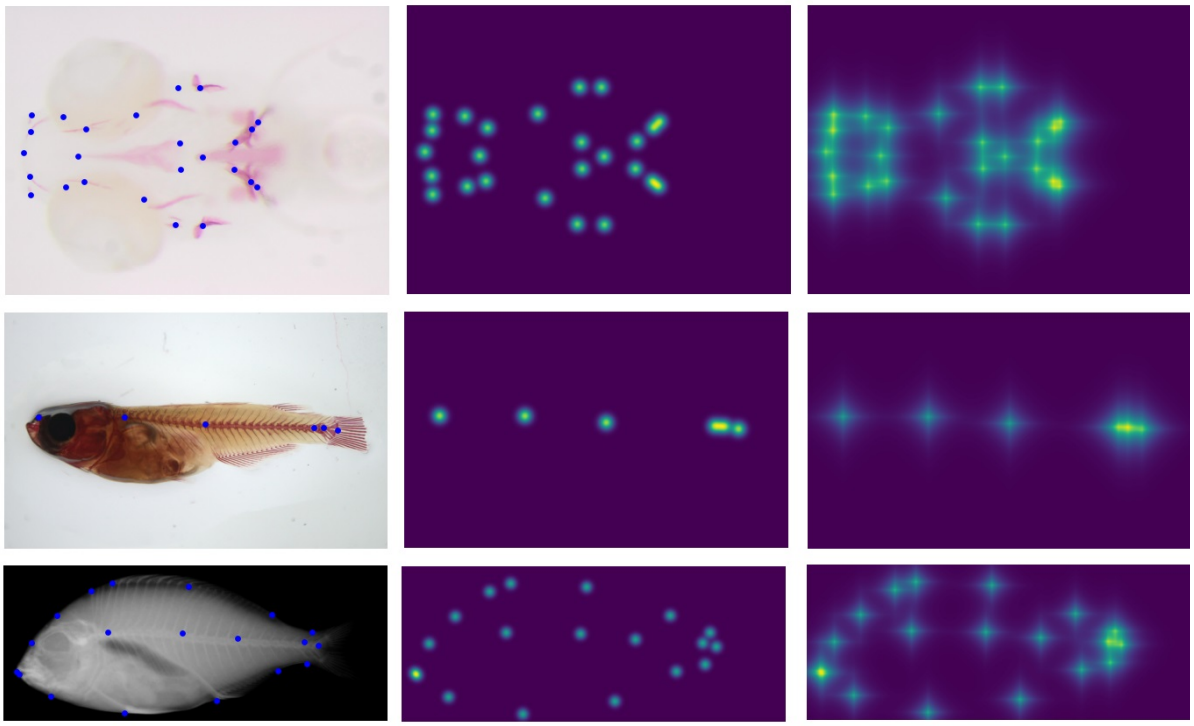

Figure 5.4: Original landmarks on the images *(first column)*, their corresponding Gaussian heatmaps *(second column)* and Exponential heatmaps *(third column)*

For direct regression, the output of the model consists of $N \times 2$ real numbers, with N the total number of landmark, representing landmark coordinates rescaled between 0 and 1. For heatmap regression, the output is composed of N heatmap slices, each corresponding to one landmark and constructed as described in the previous section.

The **prediction phase** for direct regression based approach is simply predicting the $N \times 2$ numbers and then upscaling them to the original sized image (i.e., multiplying them by the original image width and height after padding is removed). In the case of the heatmap based approach, heatmap slices are first predicted by the network and then, as a post processing step, each heatmap is converted to its corresponding landmark location by taking the *argmax* of the heatmap over all image pixel values. The *argmax* function returns the 2D coordinates of the highest value in a heatmap slice. The corresponding landmark coordinates are then upscaled to the size of the original image to produce the final model predictions.

5.4.4 Network Architectures

To evaluate our methodology, we implement state-of-art CNNs used in various image recognition, segmentation, and pose estimation tasks. Following are the CNN architectures we implement in both the multivariate and the heatmap regression based output network models.

- **Heatmap based CNN architectures:**
  - *U-Net architecture*: U-Net architecture as described in 2.4.4.3 is a two phase encoder and decoder network in which the encoder module is made up of conventional stack of convolutional layers followed by max-pooling layer and the decoder module consists in a stack of up-sampling layers. In this work, the last layer of the network is modified to output the N heatmaps.
  - *FCN8 architecture*: In FCN8 as described in 2.4.4.2, the initial layers are made up of stack of convolutional layers followed by maxpooling whereas later layers are upsampling layers that consist in the fusion of intermediate convolutional layers as shown in Figure 2.12. In this work, the last layer is modified to output N probability heatmaps.
  - *ResNet50 backbone*: ResNet50 is a state-of-the-art image recognition CNN model described in 2.4.4.1 and also successfully used in pose estimation [127]. It is made up of deeper convolutional layers with residual blocks and is capable of solving the vanishing gradient problem in deeper networks by passing the identity information in the subsequent layers. We use the upsampling layers in the decoder part to achieve the same resolution as that of the input size. We use ResNet50 pretrained on 'ImageNet' [49] dataset for our evaluation methodology (see section 2.5 for transfer learning). Note that heatmap-based regression with this architecture is very close to the DeepLabCut[85] approach and, thus, can be considered as a reimplementation of this latter method.
  - *HRNet*: The deep High Resolution Network architecture is one of the state-of-the-art architectures for the task of human pose estimation [183]. It maintains the high resolution from end to end and uses other subnetworks in parallel to exchange information between and within the stages (see Figure 2.14).
- **Multivariate regression based CNN architectures:** To implement multivariate regression that directly regresses coordinate points, we investigate two types of strategies. In the first case, the encoder part of the U-Net architecture is used for

learning feature representations. In the second scenario, we explore a transfer learning based approach where a ResNet50 network pretrained on ImageNet is used for learning representations. In both scenarios, a fully connected layer is added at the end of the network to output $N \times 2$ numbers that correspond to (x, y) coordinates of each landmark location, where N is the total number of landmark locations.

5.4.5 Experimental Protocol and Implementation

To evaluate method variants, we follow a 5-fold cross validation scheme in which each dataset is divided into 5 equal parts. In each iteration, one part is used as test set while the other four parts are merged and shuffled and used as training and validation sets, with a 3:1 ratio. Here the validation set is used for choosing the best model from the number of epochs during training. In each fold, one model is trained for maximum upto 2000 epochs. Mean error is then measured as first upscaling the predictions to the original sized images then taking the Root Mean Square Error (RMSE) (i.e., the Euclidean distance) between original ground-truth landmark locations and upscaled predicted locations for each test image, then calculating the mean over all the test images. The final error is reported by taking the mean error and standard deviation (Std.) over 5-fold cross validation. In all the evaluation protocols, we applied RMSProp optimizer with initial learning rate as 0.001 and Mean Square Error (MSE) as the loss function. To induce variability in the training set, we use data augmentation (scale, shift, rotate, shear, horizontal flip, random brightness, and contrast change) for all methods. We also use some callbacks such as *Early stopping* in which training is stopped when the loss does not improve over 400 epochs and *Learning rate scheduler* in which learning rate is reduced by the factor of 0.2 if validation loss is not improving over 200 epochs. We use *Tensorflow* [3] as the deep learning library and Python as programming language. We have trained the CNNs models on a cluster of roughly 100 NVIDIA's GeForce GTX 1080 GPUs.

5.5 Results and Discussion

Baseline: We evaluate a first baseline, called *'Mean model'*, that simply predicts for each landmark the mean positions computed for each landmark over original sized images of the training and validation sets. In Table 5.2, we report the mean error (and standard

deviation) of this model across 5-folds for our three datasets. As expected, the errors are very high, showing that landmarks positions are highly variable given the uncontrolled positioning and orientation of the fishes.

Table 5.2: Mean RMSE for 5-fold cross validation for the baseline *'Mean model'*

Dataset	**Mean error±S.D.**
Zebrafish Microscopy	77.54±8.74
Medaka Microscopy	184.96±19.11
Seabream Radiography	50.14±1.27

Direct multivariate regression: Mean errors and standard deviations over 5-fold cross validation scheme for direct multivariate regression are reported in Table 5.3. As expected, very significant improvements can be obtained with respect to the Mean model. The only exception is U-Net on the Zebrafish Microscopy dataset that obtains a higher error than the baseline. We hypothesize that this could be due to the significantly lower number of images (113) in this dataset and the fact that U-Net, unlike ResNet50, is not pretrained, which makes this model more difficult to train. U-Net remains however a better model than ResNet50 on the other two, larger, datasets.

Table 5.3: Mean RMSE for 5-fold cross validation for direct multivariate regression

Dataset	**Mean Error±S.D.**	
	U-Net(31M)	ResNet50(30M)
Zebrafish Microscopy	121.24±5.38	26.31±6.42
Medaka Microscopy	16.65±2.35	20.44±7.61
Seabream Radiography	7.71±0.2	9.65±2.34

Heatmap regression: Heatmap regression requires tuning an additional hyper-parameter, the dispersion σ. We carried out some preliminary experiments on the Zebrafish Microscopy Dataset to analyse the impact of this parameter with both heatmap generation strategies. Table 5.4 shows how the RMSE error, estimated using the validation set of a single dataset split, evolves with σ in the case of the U-Net architecture. The best performance is obtained with $\sigma = 5$ with the Gaussian heatmap and $\sigma = 3$ with the Exponential heatmap. We will therefore set σ to these two values for all subsequent experiments. This will potentially make our results on the Zebrafish Microscopy Dataset

a bit positively biased but we expect this bias to be negligible as the errors in table 5.4 remain very stable and essentially independent of σ as soon as σ is higher than 3. Note also that better results can be potentially obtained on all problems by tuning σ using some additional internal cross-validation loop (at a higher computational cost).

Table 5.4: Effect of σ values using Zebrafish microscopy validation data with U-Net

σ	**RMSE Error (in pixels)**	
	Gaussian	*Exponential*
1	1202.64	118.87
2	1417.18	1198.1
3	36.38	19.35
4	20.66	19.76
5	19.23	20.06
6	23.52	19.64
7	20.73	19.68
8	19.58	19.58
9	20.15	20.73
10	20.47	20.11

Table 5.5: Mean Error (in pixels) from 5-fold cross validation for heatmap regression

Heatmaps	**Datasets**	**Mean Error ± S.D.**			
		U-Net(31M)	FCN8(17M)	RestNet50(51M)	HRNet(6.5M)
Gaussian	Zebrafish Microscopy	13.43±3.14	13.82±2.01	13.77±2.97	13.16±2.93
	Medaka Microscopy	10.36±2.45	10.56±1.85	10.18±1.17	10.69±2.52
	Seabrean Radiography	5.69±0.28	5.74±0.15	6.13±0.31	6.40±0.63
Exponential	Zebrafish Microscopy	11.29±0.84	14.28±2.35	13.08±3.24	12.62±2.66
	Medaka Microscopy	9.34±1.06	10.12±1.60	9.36±1.05	9.54±1.59
	Seabream Radiography	5.31±0.13	5.70±0.16	5.47±0.18	5.90±0.64

Table 5.5 reports the performance of the different architectures, with both Gaussian and Exponential heatmaps. We observe that CNNs having more parameters tend to perform better in most of the cases (except HRNet with gaussian heatmap) but at the cost of computational efficiency and memory requirements. In particular, **U-Net** is better in terms of accuracy though second largest in size. Pretrained ResNet50 comes next with comparable performance with the largest size among all the models. Exponential heatmap outperforms Gaussian heatmap in almost all situations, although the difference is not very significant.

Comparing Table 5.5 with Table 5.3, it can be observed that heatmap based regression clearly outperforms direct multivariate regression on all datasets. From this investigation, we can conclude that, for the problem of landmark detection in Fish bioimages at least, heatmap based regression, with U-Net and Exponential heatmap, is the preferred approach, especially when the dataset is small.

It is interesting to note that because of the downscaling of the input image and the upscaling of the predictions, one can expect that the reported errors will be non zero even if the heatmap is perfectly predicted by the CNN model. We can thus expect that our results could be improved by using higher resolution images/heatmaps, at the price of a higher computational cost.

Hit rate: To further measure the performance of the model in terms of how many landmarks are correctly predicted, we define a prediction as a **hit** if the predicted landmark location is within some tolerance distance δ from the actual landmark location. The **hit rate** is then the percentage of landmarks in the test images that are having a hit. We choose the best performing method from Table 5.5 (exponential heatmap based U-Net model) and hit rates with different distance thresholds, estimated by 5-fold cross-validation, are shown in Table 5.6, with the baseline δ set at the ratio between the original and heatmap resolutions. As expected, there are not many hits at δ, except on the third dataset. At $2 \times \delta$ however, all landmarks are perfectly detected, which suggests that heatmaps are very accurately predicted (2 pixels error in the downscaled resolution) and further supports the idea that better performance could be expected by increasing the resolution of the network input images and heatmaps.

Table 5.6: Hit rate from the three dataset using best performing models

Dataset	δ **(in pixels)**	**Hit rate** (in %)	
		δ	$2 \times \delta$
Zebrafish Microscopy	10	20.0	100
Medaka Microscopy	10	16.66	100
Seabream Radiography	8	94.73	100

Per landmark error: To further assess performances hence derive guidelines for practical use in real-world application, we computed mean error per landmark on test sets across 5-folds in order to quantify which landmarks are hard to predict by the models. Figure 5.5 shows per landmark mean error using the best performing method (exponential heatmap based U-Net model) for all the three datasets. We can observe

that in the case of the Zebrafish Microscopy dataset, landmarks 4, 16, and 21 are the most difficult to predict. We hypothesized that these points are largely influenced by their position on the structure which they marked on. These structures exhibit some variability (shape, thickness, overlapping, missing or partially missing). In the case of Seabream Radiography, landmarks G, M, and T are difficult to predict due to their position which is somehow matched with background (see Figure 5.3). Lastly, in the case of the Medaka Microscopy dataset, landmark 3 (see Figure 5.2) is badly predicted. That might be attributed to the variability of the position it is marked on. As model predictions might vary greatly between landmarks, we believe these approaches should be combined with user interfaces for proofreading to make them effective. In practice, experts would mostly need to focus and proofread badly predicted landmarks, an hybrid human-computer approach which is expected to be much less time consuming than a completely manual approach.

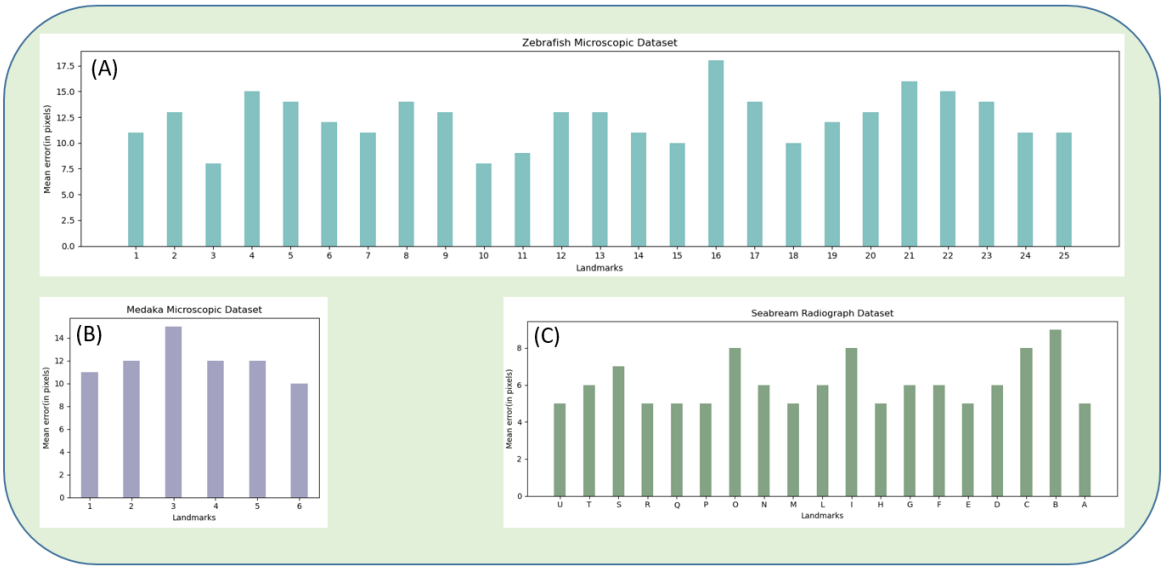

Figure 5.5: Mean error per landmark with Exponential heatmap regression based U-Net on Zebrafish (A), Medaka (B), and Seabream (C) datasets

Finally, in Figure 5.6, we illustrate the predictions from the best models using one image from the test set of each dataset.

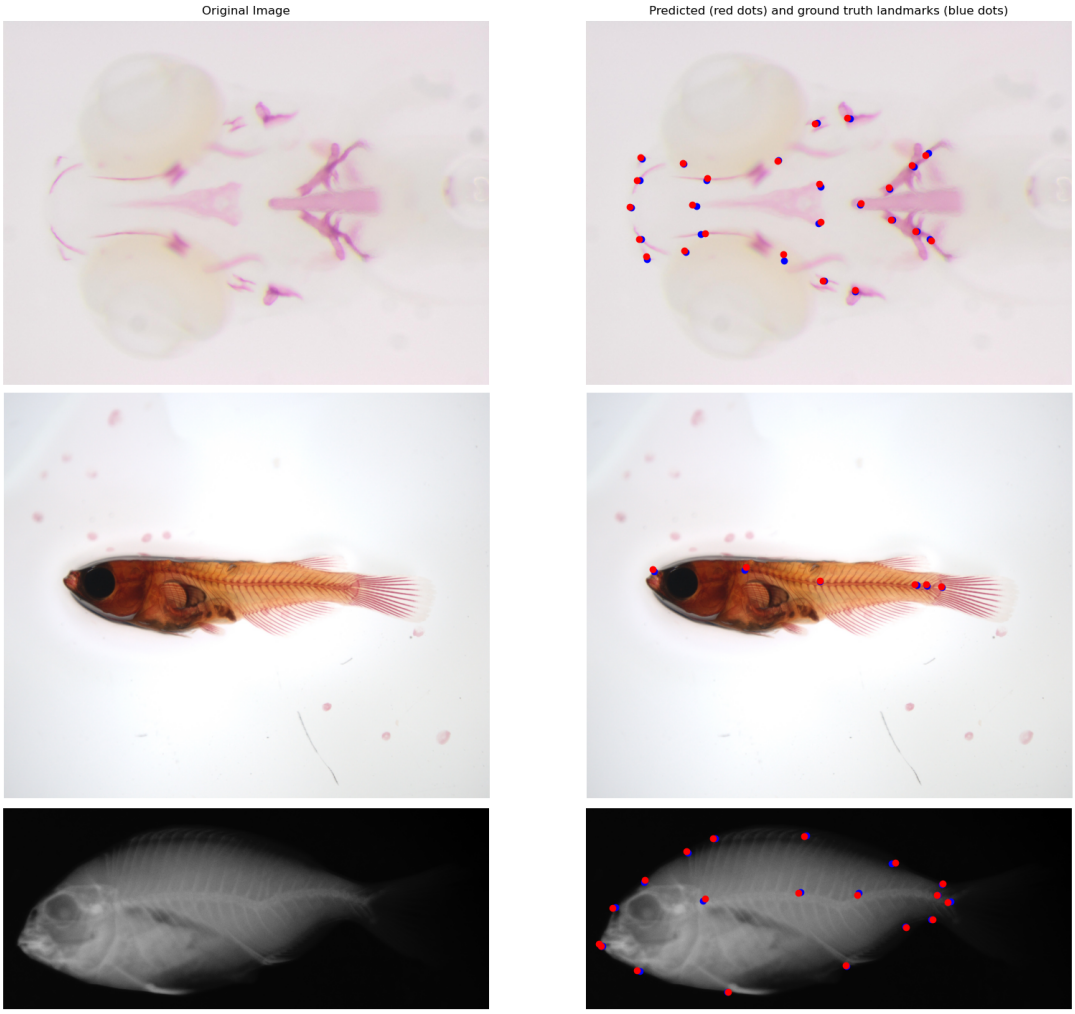

Figure 5.6: Sample predictions on one image from each of our three datasets (Zebrafish, Medaka and Seabream) using best performing models (exponential heatmap based U-Net). First column: Original image. Second column: image with predicted landmarks (red dots) and ground truth landmarks (blue dots)

5.6 Conclusions

We have evaluated two types of regression based landmark detection strategies combined with four CNN architectures on two microscopy and one radiography imaging datasets of different types of fish species with limited ground truths. The winning strategy (heatmap-based regression with Exponential generation function and U-Net architecture) is a simple end-to-end deep learning methodology where a single model is able to predict all the landmarks in a single run. Datasets and codes are distributed using open

licenses and integrated into Cytomine [124][1]. End-users can train models and proofread model predictions, then export all statistics for their morphometric studies. Preliminary experiments have showed that this approach works also well on images of butterfly wings (http://hdl.handle.net/2268.2/14509) and we expect our work will ease landmark detection in future bioimaging studies.

[1]Code: https://github.com/cytomine-uliege. Demo server with datasets: http://research.uliege.cytomine.org/ username: eccv2022bic password: deep-fish

CHAPTER 6

Uncovering the Bone Structures in Zebrafish Larvae: A Deep Learning Approach in Microscopy

As discussed in Chapter 1, the zebrafish (Danio rerio) is widely regarded as an ideal model for studying vertebrate biology in biomedical research. Its transparent body during larval stages and a genetic similarity of over 75% with humans make it especially useful for genetic and molecular studies focused on bone biology. Bone structures in developing zebrafish (at $9-10$ dpf) are typically observed by staining wildtype/untreated and mutant/treated larvae with calcium-binding dyes, like alizarin red or calcein, followed by microscopic imaging. Examining the development of bone structures in zebrafish larval microscopy images is essential to accurately analyze skeletal development and bone-related anomalies. Missing structures or gaps in imaging data, whether due to imaging limitations, staining inconsistencies, or structural irregularities from mutations can hinder accurate assessment of bone phenotypes. Deep learning techniques, specifically in image segmentation, can offer promising solutions for identifying and even reconstructing missing and occluded bone structures in zebrafish larval images. Using a deep learning model for semantic segmentation can enable precise identification and segmentation of bone regions across microscopy images, even when parts of the structure are weak, faint or occluded. In this chapter, we present a deep learning-based semantic segmentation approach to uncover the missing, weak, faint and overlapping bone structures from two microscopy image datasets of 9-day post-fertilization (dpf) zebrafish larvae, acquired

from **lateral** and **ventral** views respectively.

6.1 Introduction

Skeletal development in zebrafish is a dynamic and tightly controlled process, with individual elements developing in a predetermined sequence and timing [44, 54, 97]. Most studies in zebrafish focus on the head skeleton, as it is the first to mineralize, and the most prominent structure to be analyzed is the operculum, later covering the gills [103, 189]. However, assessing the entire head skeleton reveals that individual elements may respond differently to a stimulus [5, 6], thus assessment of the entire cranial skeleton is required. Key phenotypic outcomes include overall changes in mineralization levels [43, 150, 195] and deformities or absence of specific bone elements resulting from disruptions in morphogenic pathways [13, 28, 31]. Inter-individual variability, as well as variability in experimental conditions and timing require that comparisons between mutant/treated larvae with their wild type/untreated controls are always performed in parallel and on a sufficient number of animals. These studies provide insights into the molecular basis of bone diseases, aiding in diagnosis, and facilitating drug screening. Such research contributes to finding improved treatments for conditions associated with aging, including osteoporosis, osteopetrosis, osteoarthritis, and various bone injuries [40]. Analyzing developing bone structures from microscopy images of control and mutant/treated larvae is a crucial but time consuming task. In the process of analysing the bone structures, experts need to visually inspect each bone structure meticulously in order to specify the presence or absence of the bone structure, and its shape and size. Especially in ventral view, several structures are overlapping in 2d images (see Figure 6.1 (D)) and it is challenging to delineate the boundaries of each structure manually. Moreover, certain structures are either absent or have very unclear/weak boundaries (see Figure 6.1 (B) and (C)), posing challenges for biologists in objective visual observation. Nowadays, biomedical researchers are taking the assistance of computer vision based automatic image processing tools to reduce human error and streamline the time consuming manual annotations of the developing bone structures in the model fish.

In Section 6.2, we describe the image acquisition process and the image datasets used in our approach. Section 6.3 is dedicated to specifying the methodology, network architecture and evaluation protocol used and we present our results in Section 6.4.

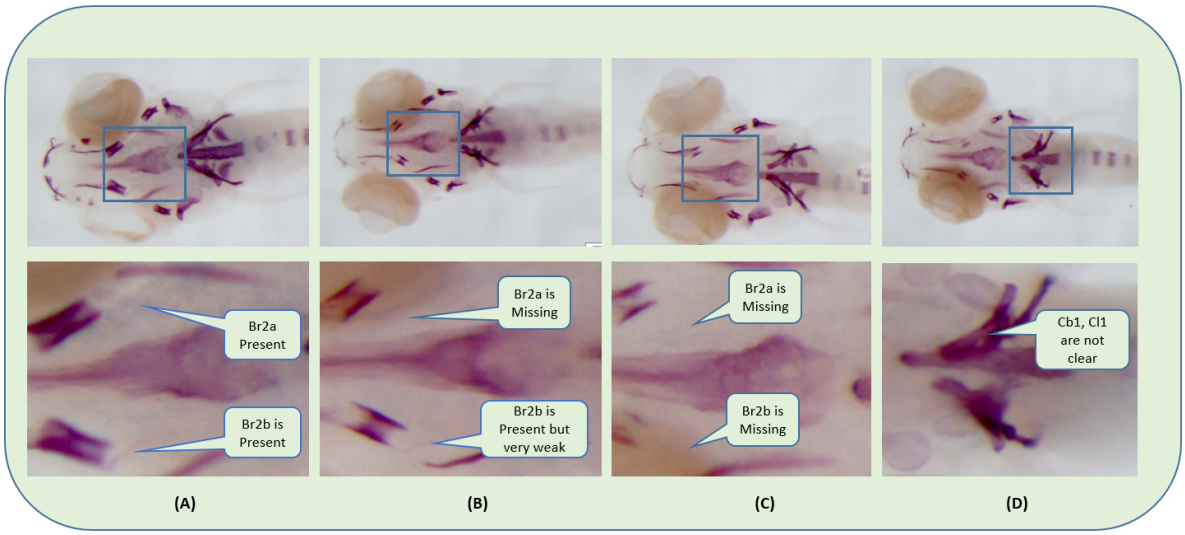

Figure 6.1: Image samples of zebrafish larvae depicting cases of present, missing, blurred, and occluded structures. The top row displays the original images, while the bottom row shows a magnified section (indicated by a blue square). In column (A), both Br2a and Br2b structures are visible. In column (B), Br2a is missing, while Br2b is faintly visible. In column (C), both Br2a and Br2b are absent. In column (D), Cb1 and Cl1 are overlapping with unclear boundaries.

6.2 Image acquisition and dataset description

In this work, we use two types of datasets, acquired using the same microscopy setting, but different views of the alizarin red stained, 3-dimensional head skeleton, namely a "lateral view" and a "ventral view". In the lateral view, euthanized larvae are placed in a side view such that the eyes and most symmetrically paired elements overlap, while the vertebral bodies are clearly visualized (see Figure 6.3 (a)). In the ventral view, the larvae are placed horizontally with the bottom (ventral) facing towards the objective of the microscope. In that view, all elements of the head skeleton are clearly observed, albeit some are again overlapping (see Figure 6.3 (b)). In the original study [150], two different mutant lines were used, carrying insertion mutations in the *col10a1a* (zfin Id: ulg076) or the *fbln1* (zfin Id: ulg075) gene coding regions, respectively, that inactivate the encoded protein. Only images from the *col10ala* mutant line is used in this chapter. For each line, three genotypes were obtained by crossing heterozygous mutant parents: homozygous (hom) carrying both copies of the mutant alleles, heterozygous (het) carrying one copy each of the mutant and WT alleles, and WT carrying only the intact alleles (controls). All larvae were sacrificed at the same age of 9 days post-fertilization (dpf), stained for

calcified structures by alizarin red, and imaged as described previously in [150] using a dissecting microscope (Olympus SZX10, Tokyo, Japan, cell B software version 3.4). Annotations were carried out by experts, while genotyping of all individual larvae was performed after all image analysis was finalized as described in [150]. Two datasets were formed: the "lateral view" dataset and the "ventral view" dataset. In the lateral view dataset, visible vertebral bodies are annotated, whereas in the ventral view dataset, all visible bone structures of the head of the zebrafish larvae are annotated. Since in both datasets, some structures are either missing or not clearly visible, the total number of structures varies from image to image.

Figure 6.2 shows sample images from both datasets and their corresponding masks. In both cases, the original image resolution is 1932×2576. The lateral view dataset contains 117 images, and the ventral view dataset contains 192 images. Images in the lateral and ventral dataset originate respectively from 36 and 38 different fish. For each fish, multiple images (on average 3 and 5 per fish, respectively for the lateral and ventral views) are present in the dataset that correspond to different views of the fish in terms of focus or orientation. Some images have also been annotated several times by different annotators. We decided to incorporate them all in the dataset, as they reflect natural diversity in the data collection protocol. We took care, however, of not incorporating images from the same fish in both the training and test set to avoid any bias in the evaluation (see Section 6.3.4 for the experimental protocol).

6.2.1 Annotation description

In the lateral view dataset, visible bone structures (vertebral bodies) of the vertebral column are annotated and all are termed as *"VB"*, a short form of vertebral body. In the ventral view dataset, an image has a maximum of 24 bone structures if none is missing. The names of the structures are as follows: Branchiostegal ray 1 (Br1a, Br1b), Branchiostegal ray 2 (Br2a, Br2b), Ceratobranchial (Cb1, Cb2), Ceratohyal (Ch1, Ch2), Cleithrum (Cl1, Cl2), Dentary (D1, D2), Entopterygoid (En1, En2), Hyomandibular (Hm1, Hm2), Maxilla (M1, M2), Notochord (N), Occipital (Oc1, Oc2), Opercle (Op1, Op2), Parasphenoid (P). Out of 24 structures, 22 structures are in 11 bilateral pairs, while two axial structures are single (N, P). Apart from structural annotations, we have also annotated the full fish larvae in ventral view to be used for automatic cropping in our experiments. Image samples from lateral and ventral view datasets with their corresponding annotated bone structures are shown in Figure 6.3.

While visualizing the ventral view dataset, we observe that the boundaries of some

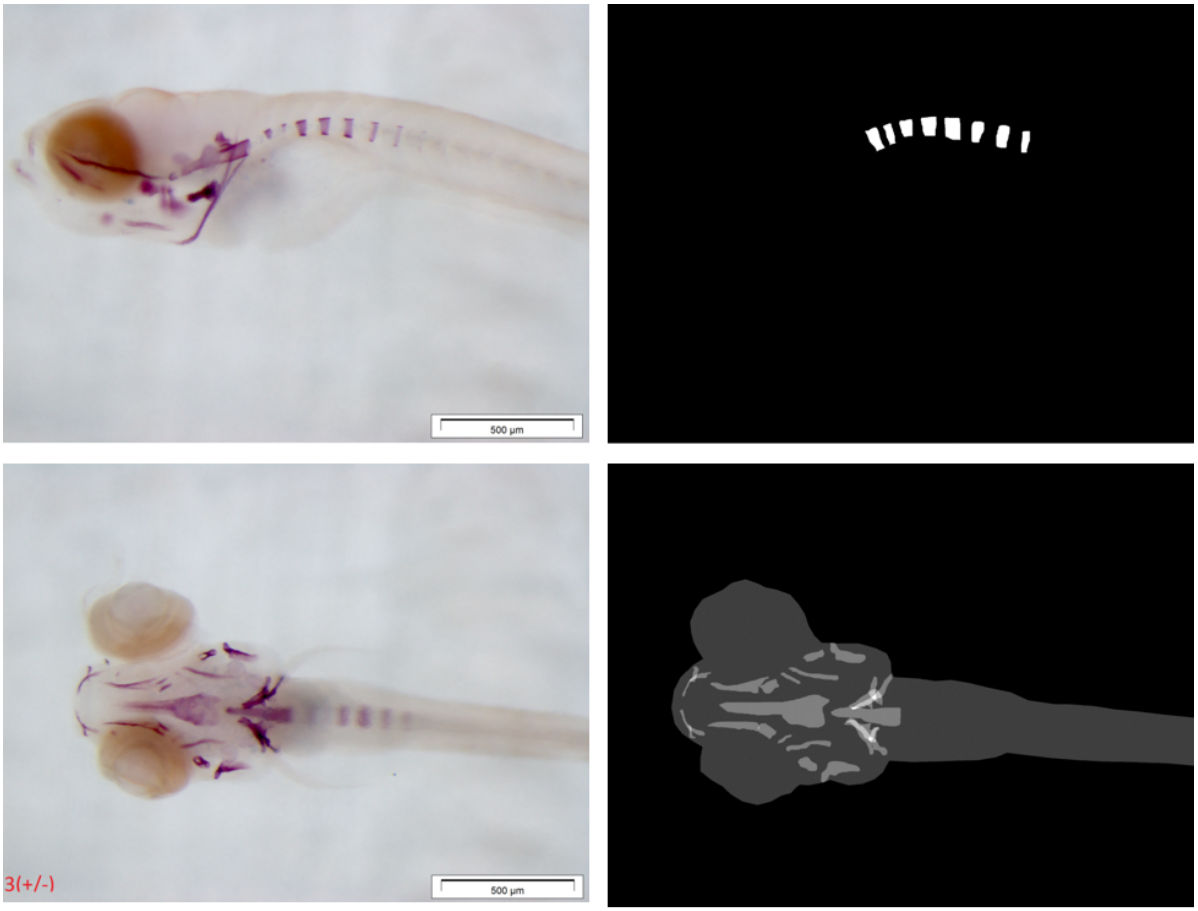

Figure 6.2: Sample images from lateral view dataset (first row) and ventral view dataset (second row) with their corresponding segmentation masks. The scale bars correspond to 500 µm.

structures are either blur or unclear due to overlapping with other structures. As shown in Figure 6.1 (B), Br2b is very weak but present and in Figure 6.1 (D), the boundaries of Cb1 and Cl1 are not clear due to overlapping. Because of these limitations in visual perception, structures may not get accurate annotations, potentially inducing subjectivity in manual annotations by the experts and resulting in the possibility of mislabeled data.

6.3 Method Description

To identify and segment missing, faint and occluded bone structures in microscopy images, we employ a "binary semantic segmentation" approach, where each pixel is classified as either positive (1) or negative (0). Clusters of positive pixels (1) represent

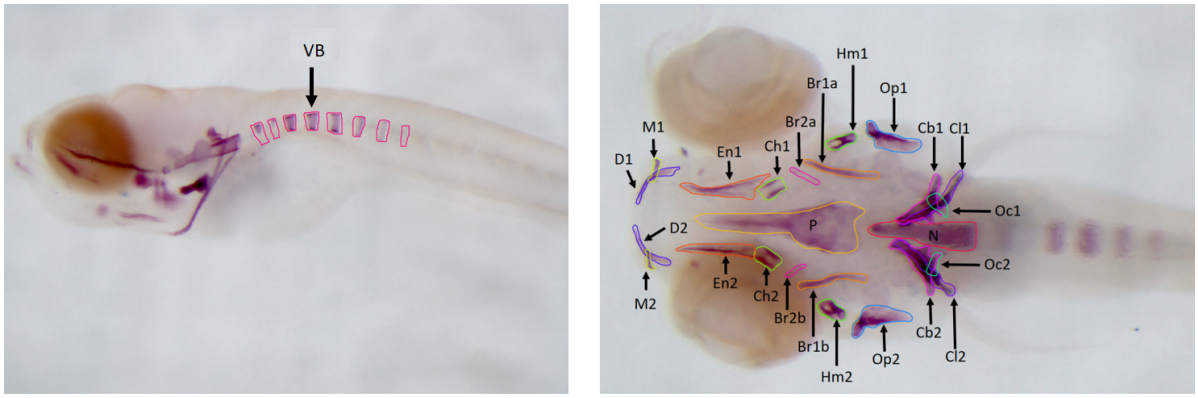

Figure 6.3: Sample images with annotations from (a) lateral and (b) ventral view dataset.

regions of interest (ROIs ie. segmented bone structures), while clusters of negative pixels (0) represent the background. We evaluate the model on test images by determining the overlap between the predicted segmented structures and ground truth masks (we discuss in detail about training and prediction phases in Section 6.3.3). Two distinct binary segmentation U-Net models, with modifications at the output layer, are trained separately for the lateral and ventral view datasets.

In the lateral view dataset, a binary segmentation U-Net model with a single output layer is trained to segment all *"VB"* structures within the vertebral region of zebrafish larvae. The use of a single output layer is appropriate due to the lack of structural overlap. Furthermore, full-size images are utilized without cropping, and they undergo resizing and padding during the training process (refer to Section 6.3.3 for details on the training phase).

For the ventral view dataset, the approach involves first training a binary segmentation U-Net model to segment the entire larval body from full-sized images. The resulting segmentation masks are then used to automatically crop the images. This cropping step, applied to the ventral view dataset is intended to increase the proportion of positive class pixels by reducing the background (negative class pixels), thereby partially addressing the issue of class imbalance (we discuss the class imbalance problem later in this section).

As described in Section 6.2, ventral images contain 24 structures, of which 22 form 11 pairs with symmetrical shapes (e.g., Br1a and Br1b or M1 and M2 are symmetrical). Two structures (N and P, shown in Figure 6.3) are unpaired, as they are located along the medial axis. Additionally, some structures overlap (e.g., D1, M1, and Cb1, Cl1, Oc1, and N, as seen in Figure 6.3), which complicates the segmentation task for a binary model

with a single output mask. Furthermore, the dataset is affected by *class imbalance*, as the average ratio of positive pixels to background pixels per image (and thus per mask) is below 0.3, a problem that could worsen if separate output masks are used for each structure. To tackle both these issues, we create the final ground truth masks by combining each symmetrical pair of structures into a single mask, while structures without symmetry are kept as separate masks. This approach helps mitigate class imbalance and addresses the overlap issue by merging symmetrical structures into one mask. We then stack these masks along the third axis to produce a multi-layer output mask with dimension $H \times W \times 13$ where H and W are the height and width of the input image, and 13 is the total number of output masks (11 paired and 2 unpaired structures). If a structure is absent in the ground truth, the combined mask contains only background pixels (all values set to 0), while any present structure is marked with positive class values (1s) in its respective locations. If all structures in a mask are missing, then we get a fully negative mask (all values as 0) along the third dimension, ensuring a consistent number of masks (*i.e.* 13) at the output layer. Figure 6.2 shows example images from both lateral and ventral views with their corresponding segmentation masks. For visualization, all ventral masks are merged into a single mask. Note that only head structures from the ventral view dataset are used in this methodology. Figure 6.4 depicts the end-to-end methodology we use for the lateral view (above) and ventral view (below) datasets.

6.3.1 CNN architecture

We implemented a CNN architecture called U-Net for both image datasets. Originally introduced in [157] (and discussed in Section 2.4.4.3), U-Net is a deep learning model designed for semantic segmentation in biomedical images. It utilizes an encoder-decoder structure, where the encoder comprises conventional blocks of convolutional layers followed by max-pooling layers. The max-pooling layers reduce the resolution of the activation maps, enhancing spatial invariance of the features. The decoder block, composed of upsampling or deconvolutional layers, restores the original image resolution. Each convolutional block's activation maps are added to the corresponding deconvolutional block, transferring feature information from the encoder to the decoder during upsampling. This transfer aids the learning process by addressing vanishing gradients. Additionally, batch normalization is applied throughout both modules to enable more efficient network training. Figure 6.5 shows the exact U-Net architecture used in our methodology.

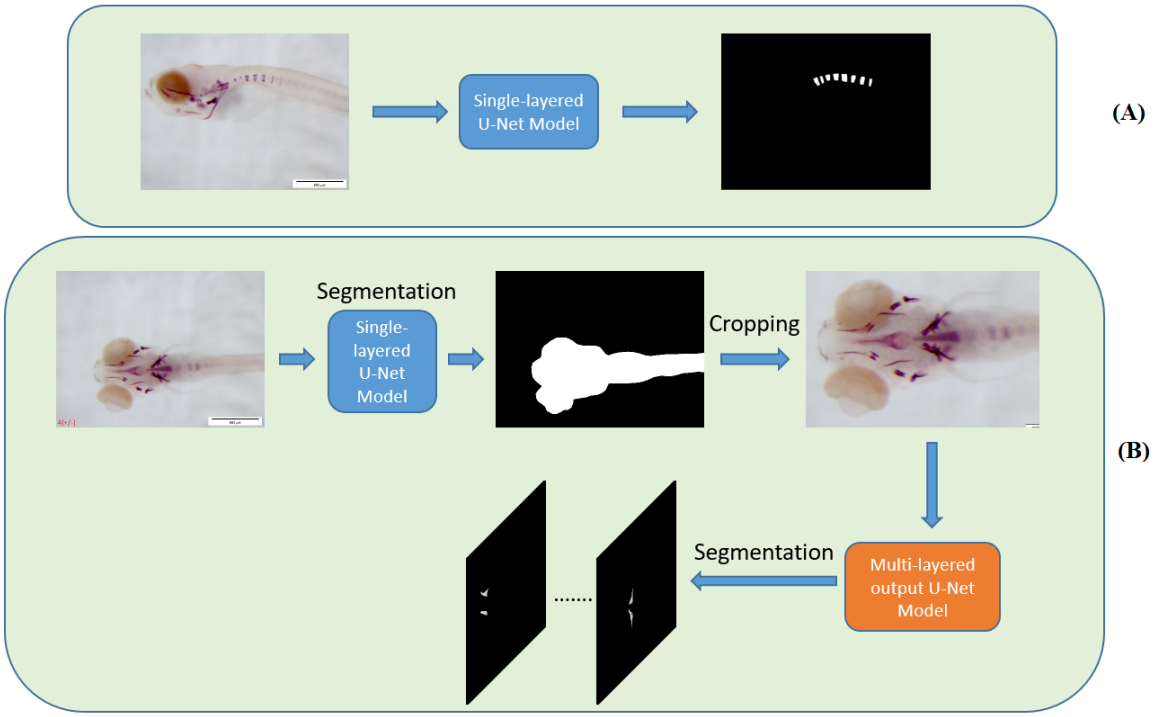

Figure 6.4: Method description for lateral view (A) and ventral view (B) dataset training

6.3.2 Loss functions

To handle the problem of class imbalance, we use 'focal loss' as CNN's cost function. Focal loss is a generalized version of cross-entropy (CE) loss that tries to focus more on 'hard to classify' (FN) examples and down-weigh the 'easy to classify' (TN), thus reducing the overhead of the *class imbalance* problem. To investigate the robustness of the model against "hard to annotate" structures (*e.g.* Br2a, Br2b, M1, Cb1, Cl1 etc.) in some images, we experiment with a model using 'bi-tempered logistic loss' combined with 'focal loss' (see Section 2.6.2 for detailed description about these loss functions).

6.3.3 Training and prediction phases

In the **training phase**, original images are first downscaled to 512×512 to be fed into the network for both datasets. Since the original images are rectangular, we first downscale the image to a size of 512 along the largest dimension while keeping the aspect ratio unchanged. Padding (with zeros) is then added to the smallest dimension to produce a 512×512 square image. Following this procedure, in lateral view dataset, the output of the model consists of a single mask of dimension $H \times W$ with H and W corresponding to

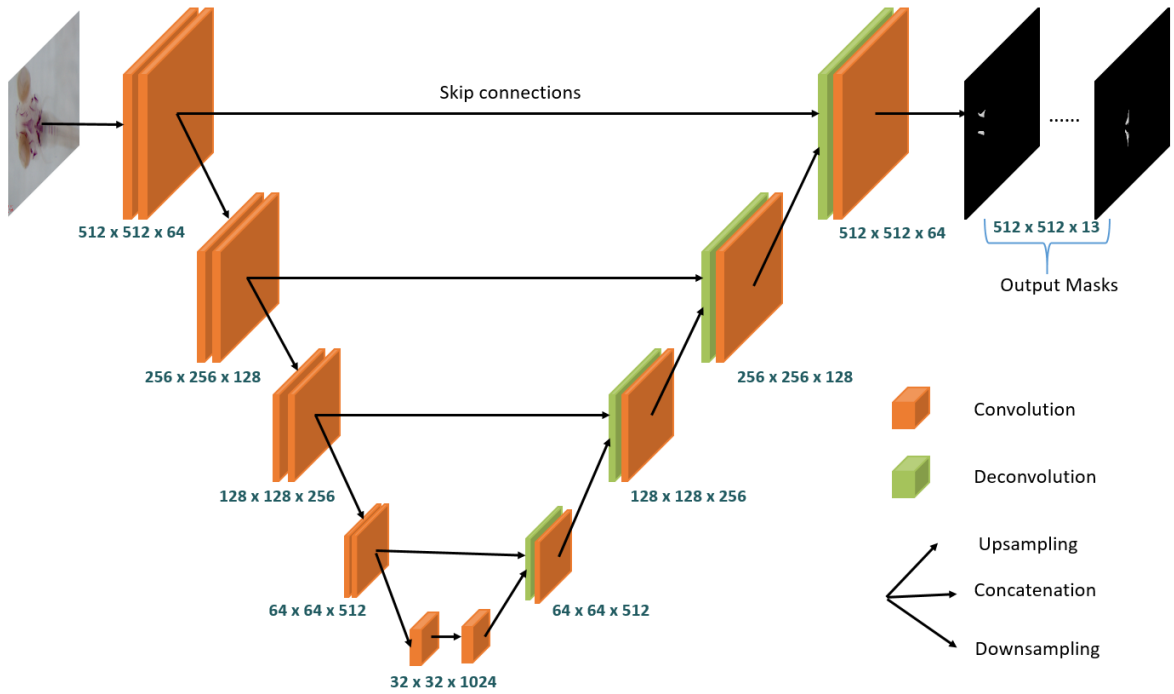

Figure 6.5: UNet architecture (with modifications at last layer) used in our experiments

the height and width of the input image respectively, while in ventral view dataset, the output of the model is composed of 13 masks with height and width equal to those of the input image.

In the **prediction phase**, all the *VB* structures are predicted with a single output mask in case of lateral view dataset and then this mask is upscaled to the original image size to be compared with the original ground truth masks. This upscaling is performed by first removing the padding, then resizing it to the original height and width of the image. The same upscaling steps are applied to upscale all the output masks in the ventral image dateset as well. In ventral view dataset, the third dimension (depth) of the output mask is 13, hence each slice of the output mask either corresponds to paired structures or single structure. During prediction phase, we may get some blobs of false positives along with ROIs (*i.e.* true positives) in the slices of the predicted mask in both the lateral and ventral view images. For mask slices having single structure (*i.e.* P and N), in our experiments blobs of false positives are removed by considering only the potential predicted structures that have pixels count of more then 70 (as it was observed on our dataset that P and N structures have more than 1000 pixel average area). False positives blobs from paired structure slices of the predicted mask are removed by considering only the blobs for potential predicted structures that have more then 25 pixel count.

6.3.4 Experimental protocol and implementation

To evaluate the models in both image datasets, we adopt a K-fold cross validation strategy where the dataset is partitioned into K folds of (approximately) equal sizes. In each iteration, one fold serves as the test set, while the remaining $K-1$ folds are combined, shuffled, and split into training and validation sets. Here the validation set is used to select the best model for predicting the structures. This approach allows every image in the dataset, the opportunity to serve as a test image when its corresponding fold is designated as the test set. Following this rule, we choose the value of $K = 5$ for both the lateral view and the ventral view dataset. Care was taken to put all the images of the same fish in the same fold to avoid any bias in the evaluation. For each fold training, the model is trained for 2000 epochs and is saved at the current checkpoint (using checkpoint callback) if the loss at current epoch is improved over the loss at previous epoch. Model training on the current fold is stopped, and moves to next fold, if the training at the current fold does not see any performance improvement in validation loss for the next 300 epochs (using early stopping callback). We select precision, recall and dice score as metrics for the evaluation of the model. The Dice score represents the area of overlap between the predicted structures and the ground truth masks and ignoring the background, which is also equivalent to the $F1$ score (see Section 2.6.1.2). It directly assesses how much the predicted and actual areas align, without taking into account the background pixels. Precision measures the fraction of pixels predicted as positive by the model that are truly positive (True positive/(True positive + False positive)). Recall measures the fraction of truly positive pixels that are correctly predicted by the model (True positive/(True positive + False negative)).

In case of the **lateral view dataset**, we first calculate precision, recall and dice score per structure at the ***pixel level*** and then average them to compute the average precision, recall and dice score per test image. The final precision, recall and dice scores are computed by averaging them over all the folds. We then quantify performance at the structure and image level. To reduce tiny false positives in the predictions, we establish a criterion wherein regions (*i.e.* blobs of predicted positive pixels) within the predicted mask must contain a minimum of 25 pixels to be considered a potential candidate for structure predictions. We choose this values as there is no structure which has an area of less than 50 pixels in the original full size ground-truth masks. We first determine ***image level accuracy***, which is calculated as the proportion of images in which all ground truth structures are correctly predicted by the model, relative to the total number of images in the dataset. A ground truth structure is deemed predicted correctly if the

dice score is at least 0.5 with one of the predicted structures. Next, we compute the ***structure level accuracy***, defined as the proportion of the ground truth structures across all images in the dataset that are correctly predicted.

For the **ventral view dataset**, we first train a single output layered U-Net with a full larval body mask in order to automatically crop the images around the body of the zebrafish larvae. After creating the cropped dataset of ventral images, we proceed with the training and evaluation of the multi-layered output version of the U-Net architecture. We first compute the average precision, recall, and Dice score (*i.e.* $F1$ score) score for each structure at ***pixel level*** across 5 folds. Then, we also quantify the performance at the image level. As for the ventral view dataset, we set a criterion that a region (blob of positive pixels) must contain more than 25 positive pixels to be considered a candidate structure for prediction. The selection of this value is based on the observation that the structures in the ventral images do not contain fewer than 70 pixels (40% of $70 \approx 25$). We then compute the number of false positive and false negative predictions at the image level for each structure separately. A false positive is an image where the structure is missing in the ground truth but there is at least one predicted structure somewhere in the image. A false negative is an image where the structure is present in the ground truth but no predicted structure has a dice score above 0.4 with respect to that structure. A lower Dice score threshold is used for the ventral view due to the subjectivity in expert annotations for certain structures that may appear weak, blurred, or overlap with other structures. The overall ***image level accuracy*** for a given structure is then the percentage of images that are neither false positive nor false negative.

6.4 Results and Discussion

We first evaluate our model on the "lateral view" dataset using **5-fold cross validation** without cropping. In Table 6.1, we report at pixel level, the average precision, recall and Dice score with standard deviation (S.D.) across 5 folds. Next, we proceed with computing the accuracies at image and structure level mentioned in Section 6.3.4. They are reported in Table 6.2.

Following the protocol mentioned in Section 6.3.4, we evaluate our multi-output mask U-Net model using 5 fold cross-validation on the "ventral view" dataset. The average precision, recall and Dice score ($F1$ score) for each structure at 'pixel level' across 6-folds are reported respectively in the second, third and fourth columns of Table 6.3. The total number of missing structures in ground truth images is mentioned in the fifth column

Table 6.1: Pixel-level precision, recall and dice score, averaged over all test images, using 5-fold cross validation on the lateral view dataset.

Metric	**Score ± S.D.**
Precision (at pixel level)	0.8658 ± 0.027
Recall (at pixel level)	0.8382 ± 0.018
Dice score (at pixel level)	0.8494 ± 0.023

Table 6.2: Accuracy at image level and accuracy at structure level for the lateral view dataset, using 5-fold cross-validation.

Metric	**Score**
Accuracy (at image level)	0.88
Correct predictions (Out of 117 images)	103
Accuracy (at structure level)	0.97
Correct predictions (Out of 664 structures)	648

of Table 6.3. Next, we report in the same table the number of false positive and false negative images and the image level accuracy, as described in Section 6.3.4.

Discussion. From the results in Table 6.2, we infer that for lateral view dataset, our model performed well in predicting the **VB** structures from the tail part of the fish. By visually inspecting the predictions, we observed that our model did not predict some structures which are present in ground truth annotations primarily due to the fact that these unpredicted structures are only very weakly visible in the images. In some of these cases, different annotators might reasonably have omitted marking the vertebral bodies that our model missed (see Figure 6.6). Irrespectively of the quality of the prediction of individual structures, the model correctly predicted the exact number of vertebral bodies in 101 of the 117 images (86.3%). In the remaining images, the difference between the predicted and the actual counts was minimal: 11 images showed a difference of 1, 4 showed a difference of 2, and 1 image showed a difference of 3. With only one exception (the image with a difference of 3), all counting errors were underestimations. Given the limited size of the dataset, we believe these results are very satisfying.

For the ventral view dataset, Table 6.3 shows that structure 'P' is predicted with 100% accuracy across all folds. This high accuracy is due to its large area, distinct boundaries,

Table 6.3: Average scores per structure (at pixel level) across 5-fold cross validation (Precision, Recall and F1 score) and number of correctly predicted (as present or missing) for ventral view dataset

Structures	Precision ± S.D.	Recall ± S.D.	F1 score ± S.D. (Dice score)	Number of missing structures	False positives	False negatives	Correct predictions (out of 192)	Prediction accuracy (in %)
Br1a	0.75 ± 0.01	0.58 ± 0.06	0.58 ± 0.07	1	1	0	191	99
Br1b	0.59 ± 0.08	0.73 ± 0.04	0.73 ± 0.02	3	2	6	184	97
Br2a	0.56 ± 0.13	0.53 ± 0.13	0.54 ± 0.13	137	5	42	145	76
Br2b	0.52 ± 0.13	0.52 ± 0.13	0.52 ± 0.13	132	3	48	141	73
Cb1	0.67 ± 0.15	0.62 ± 0.13	0.64 ± 0.14	9	9	3	180	94
Cb2	0.64 ± 0.15	0.66 ± 0.14	0.63 ± 0.14	5	5	1	186	97
Ch1	0.68 ± 0.15	0.59 ± 0.14	0.60 ± 0.14	9	5	22	165	86
Ch2	0.64 ± 0.12	0.55 ± 0.08	0.56 ± 0.07	8	1	13	178	93
Cl1	0.56 ± 0.04	0.54 ± 0.03	0.55 ± 0.03	0	0	2	190	99
Cl2	0.44 ± 0.13	0.49 ± 0.10	0.46 ± 0.11	0	0	2	190	99
D1	0.51 ± 0.04	0.51 ± 0.07	0.50 ± 0.04	0	0	0	192	100
D2	0.53 ± 0.14	0.56 ± 0.12	0.54 ± 0.12	0	0	0	192	100
En1	0.67 ± 0.11	0.66 ± 0.04	0.64 ± 0.05	0	0	0	192	100
En2	0.53 ± 0.09	0.52 ± 0.10	0.51 ± 0.09	0	0	0	192	100
Hm1	0.84± 0.07	0.79 ± 0.07	0.81 ± 0.07	0	0	0	192	100
Hm2	0.80 ± 0.07	0.77 ± 0.07	0.78 ± 0.07	0	0	0	192	100
M1	0.55 ± 0.06	0.54 ± 0.10	0.52 ± 0.08	10	1	24	167	87
M2	0.59 ± 0.08	0.53 ± 0.09	0.53 ± 0.07	5	1	24	167	87
N	0.89 ± 0.01	0.84 ± 0.04	0.85 ± 0.03	0	0	0	192	100
Oc1	0.60 ± 0.09	0.48 ± 0.06	0.51 ± 0.06	32	8	24	160	83
Oc2	0.54± 0.05	0.46 ± 0.03	0.48 ± 0.01	38	9	25	158	82
Op1	0.84 ± 0.06	0.83 ± 0.04	0.83 ± 0.04	2	2	0	190	99
Op2	0.78 ± 0.07	0.76 ± 0.05	0.75 ± 0.05	0	0	0	192	100
P	0.89 ± 0.03	0.90 ± 0.01	0.89 ± 0.02	0	0	0	192	100

lack of overlap with other structures (see Figure 6.3), and also to its presence in all images. Similarly, structures 'N', 'Hm1', 'Hm2', 'D1', 'D2', 'En1', and 'En2' also achieve perfect accuracy, mainly because they have clear, non-overlapping boundaries and appear consistently in all images. 'Op2' is perfectly predicted as well, but there are two false positives in the case of 'Op1'. 'Cl1' and 'Cl2' that are present in all images are very well predicted but are wrongly undetected in two images. Unsurprisingly, the remaining structures that have an overlap with others, small size, blurred boundaries and/or a faint presence are more challenging to detect for the model and their prediction accuracy is inversely proportional to the number of images where the structure is missing. 'Cb1' and 'Cb2' present a prediction accuracy of about 95%, 'M1' and 'M2' of about 85%. The most difficult structures are 'Oc1' and 'Oc2' (~ 80% accuracy) that have a small size and a strong overlap with 'Cb1'/'Cb2' and 'Cl1'/'Cl2' and 'Br2a' and 'Br2b' (~ 75% accuracy) that are missing in more than 70% of the images. Note that a majority of the errors are false negatives, *i.e.* structures that are present in the ground truth but not detected. The

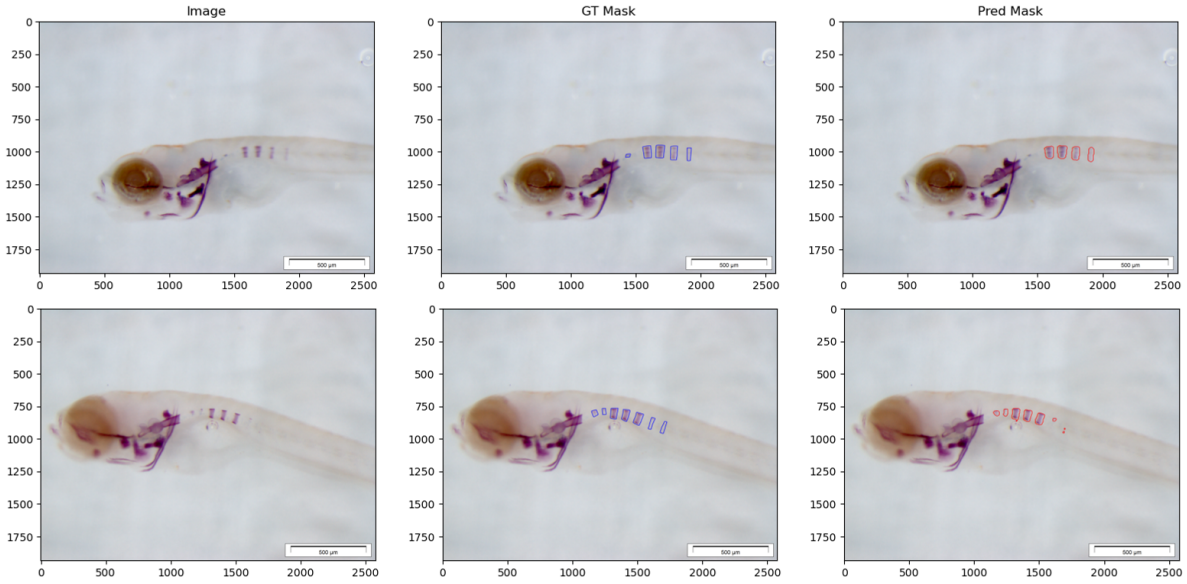

Figure 6.6: Two cases of bad predictions from the lateral view dataset. The first row shows the case where unpredicted structure is very small. The second row contains the case where unpredicted structures are slightly visible and subjectively annotated. The first column represents the original full-size images, the second column shows the ground truth annotations of the full-size images, and the third column displays the full-size predicted annotations.

Dice score at the pixel level is not perfectly correlated with the prediction accuracy at the image level, as some perfectly detected structures have low dice score (e.g., 'D1' and 'D2'). This translates the fact that some structures have blurred boundaries, but are not difficult to detect.

Impact of the genotype on bone development. One of the objectives of the original study [150] was to identify relationships between fish genotype and the presence/absence of some structures. To illustrate this downstream task, we conducted chi-square tests for each structure to detect significant dependencies between genotype and structure presence/absence. The genotype was divided into two categories (homozygous vs {heterozygous,wild-type}). We performed these tests using ground truth annotations first and then, our model's predictions obtained by 5-fold cross-validation. Table 6.4 reports p-values of both tests for all structures such that at least one of the two tests yielded a p-value below 0.05. The results reveal that the relationship between the genotype and the presence/absence of structure 'M1' is confirmed by both the ground truth and the predicted structures. The dependence of 'Oc1' is considered significant

Table 6.4: P-values for a chi-square test, performed at the **image level**, comparing the genotype with the presence/absence of a structure, with the ground truth annotations (second column) and the model predictions obtained by 5-fold cross-validation (third column). Only the structures for which at least one p-value is lower than 0.05 are shown.

Structures	P-value Ground-Truth	P-value Predictions
M1	0.000025	2.51e-11
Oc1	0.000196	0.1175249
Ch1	0.359584	0.0300473
M2	0.417581	8.43e-11

Table 6.5: P-values for a chi-square test, performed at the **fish level**, comparing the genotype with the presence/absence of a structure, with the ground truth annotations (second column) and the model predictions obtained by 5-fold cross-validation (third column). Only the structures for which at least one p-value is lower than 0.05 are shown.

Structures	P-value Ground-Truth	P-value Predictions
M1	0.008917	0.006183
Br1b	0.049183	0.851399
M2	0.231883	0.000270

when using the ground truth but it is not confirmed using the model predictions. On the other hand, the model predictions highlight a significant link between 'Ch1' and 'M2' and the genotype that is not observed using the ground truth.

Since several images are from the same fish, the p-values in Table 6.4 are too optimistic. We also performed the same tests but this time at the fish level. The genotype was also encoded into the same two categories. The structure presence/absence information was encoded into three categories: 0 if the structure is missing in all fish images, 1 if it is present in all fish images and 2 otherwise. Table 6.5 compares the p-values of both tests for all structures such that at least one of the two tests yielded a p-value below 0.05. At the fish level, only 'M1' is deemed significant by both tests. 'Br1b' is significantly linked with the genotype when using the ground truth annotations but not when using the predicted ones, while the opposite is true for 'M2'. Given the symmetry between 'M1' and 'M2', it makes sense that both are linked with the genotype and this connection was also reported in [150] based on a different manual re-annotation of the fish.

Overall, Tables 6.4 and 6.5 show that tests based on predicted structures do not lead to exactly the same conclusions as tests based on manual annotations. This discrepancy was anticipated given that model predictions do not perfectly align with ground truth

(as shown in Table 6.3). Note however that the manual annotations are not expected to be flawless either, given the difficulties discussed previously. As a consequence, we believe that the significant genotype associations identified through model predictions certainly deserve further investigation. Machine learning-based predictions may actually capture more systematic patterns than manual annotations, and, like the ground truth annotations, they are expected to be free from potential bias since they were generated without any consideration of the genotype information.

6.5 Experiments with mislabeled data

Lastly, we conduct experiments using artificially 'mislabeled/corrupted' data. The primary goal of this experiment is to evaluate the robustness of the model when trained with mislabeled images and to check the impact of different training losses on this robustness. As discussed in Section 6.1, annotations can be subjective and may result in mislabeling errors by experts due to the challenges associated with visual observations (see also Figure 6.1). In this experiment, we focus on the ventral view dataset and we deliberately corrupt it by mislabeling some structures in the images. Mislabeling is applied only to structures that are difficult to annotate, such as those that are missing in some images or have small, weak, faint, or overlapping boundaries, *i.e.* M1, M2, Br2a, Br2b, Cl1, Cl2, Oc1, Oc2, Cb1, Cb2. Perturbations involve omitting annotations for very weak but labeled structures (*e.g.* M1 and M2), and subjectively annotating structures when they overlap or are missing (*e.g.* Oc1, Oc2, Cb1, Cb2) (see Figure 6.7).

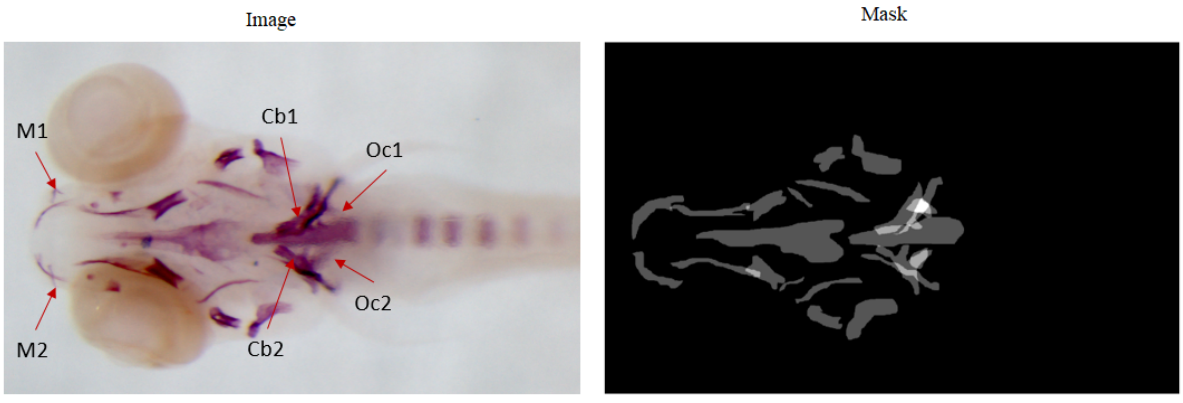

Figure 6.7: Random perturbations applied to different structures. M1 and M2 are visible in the image but omitted in the mask. Oc1, Oc2, Cb1 and Cb2 are absent in the original ground truth but subjectively annotated in the mask.

Following this protocol, we first divide the dataset in train, validation and test sets into $70:15:15$ ratio respectively. As in the previous experiments, the test set contains (30) images from (6) fish that are not present in the train and validation sets to avoid any bias. Then, we corrupt 10% of the training set while keeping the validation and test set unaltered. More precisely, for each structure in the above list, we randomly pick 10% of the training images where the structure is present and we remove it from the ground truth. We then randomly select 10% of the images where the structure is missing and we manually and subjectively add an annotation in the images at the position where the structure should have been. The model is then trained on the resulting corrupted training set for 3000 epochs in a single run using three loss functions: focal loss, bi-tempered loss, and a combination of bi-tempered and focal loss called **focally weighted bi-tempered loss**. The first two losses are described in Section 2.6.2.2. The third one is novel and detailed below. This results in three separate models, each trained with a different loss function. We also train models using the uncorrupted original training set with the same loss functions for comparison. The results are reported and discussed in Section 6.5.1.

Focally weighted bi-tempered loss. For the experiments in this section, we implement a novel loss function, called 'focally weighted bi-tempered loss', which is a hybrid version of the focal and bi-tempered losses. The basic idea of implementing this loss function is to design a loss that can handle class imbalance and mislabeling simultaneously. We define the focally weighted bi-tempered loss function for binary classification problems as follows:

$$\mathcal{L}_{FBT} = \mathcal{L}_{FL} \times \mathcal{L}_{Bi-Tempered} \tag{6.1}$$

where the focal loss $\mathcal{L}_{FL}$ is defined in Equation 2.50 of Section 2.6.2.2.5 and the bi-tempered loss $\mathcal{L}_{Bi-Tempered}$ is defined in Equation 2.54 of Section 2.6.2.2.6. The resulting loss thus depends on the tunable hyper-parameters of both losses, *i.e.* α and γ for the focal loss and the two temperatures T_1 and T_2 for the bi-tempered loss. In all our experiments, we set α to 0.8, γ to 2 (default), T_1 to 0.8 and T_2 to 1.8.

6.5.1 Results

We present preliminary experimental results with the focal loss, the bi-tempered loss, and the combination of both, using the original training set and a training set with 10% 'mislabeled/corrupted' data in Tables 6.6 and 6.7 respectively.

From Table 6.6, we can see that the focal loss performs better than the bi-tempered and the focally weighted bi-tempered losses in terms of structure predictions. This might

Table 6.6: Results on test images using various loss functions for original training dataset

Structures	Total structures	Focal				Bi-tempered				Bi-tempered + Focal			
		Dice score	Correct predictions (out of 30)	Mistakes		Dice score	Correct predictions (out of 30)	Mistakes		Dice score	Correct predictions (out of 30)	Mistakes	
				False positive	False negative			False positive	False negative			False positive	False negative
Br1a	30	0.79	30	0	0	0.81	30	0	0	0.79	30	0	0
Br1b	30	0.56	30	0	0	0.76	29	0	1	0.72	30	0	0
Br2a	17	0.60	25	1	4	0.65	24	1	5	0.66	28	2	3
Br2b	17	0.44	19	0	11	0.51	25	0	5	0.52	28	1	4
Cb1	30	0.62	29	0	1	0.59	26	0	4	0.58	27	0	3
Cb2	30	0.61	28	0	2	0.57	27	1	2	0.65	29	0	1
Ch1	30	0.81	30	0	0	0.81	30	0	0	0.81	30	0	0
Ch2	30	0.69	30	0	0	0.80	30	0	0	0.80	30	0	0
Cl1	30	0.58	27	0	3	0.59	26	0	4	0.55	26	0	4
Cl2	30	0.60	30	0	0	0.63	29	1	0	0.61	30	0	0
D1	30	0.70	30	0	0	0.70	29	1	0	0.73	29	0	1
D2	30	0.82	30	0	0	0.83	29	0	1	0.77	30	0	0
En1	30	0.72	30	0	0	0.75	30	0	0	0.76	30	0	0
En2	30	0.69	30	0	0	0.79	30	0	0	0.72	30	0	0
Hm1	30	0.84	30	0	0	0.85	30	0	0	0.81	30	0	0
Hm2	30	0.83	30	0	0	0.81	30	0	0	0.74	30	0	0
M1	28	0.62	26	1	3	0.65	28	0	2	0.53	29	1	1
M2	29	0.57	29	1	0	0.61	30	0	2	0.52	29	1	1
N	30	0.82	30	0	0	0.85	29	0	1	0.82	29	1	0
Oc1	20	0.63	30	0	0	0.56	27	3	0	0.55	27	3	0
Oc2	20	0.62	30	0	0	0.55	29	0	1	0.57	29	1	0
Op1	30	0.90	30	0	0	0.90	30	0	0	0.86	30	0	0
Op2	30	0.88	30	0	0	0.88	30	0	0	0.85	30	0	0
P	30	0.91	30	0	0	0.91	30	0	0	0.87	30	0	0
Total	**671**			**27**				**33**				**28**	

be due to the fact that bi-tempered loss is specifically built to handle mislabeled/corrupted data. Since in this case our dataset is neither mislabeled nor corrupted, at least explicitly, we might expect some errors using the bi-tempered loss and its variant. On the other hand, by looking at Table 6.7, where we use dataset with 10% mislabeling, the focal loss has more prediction errors (35 mistakes) as compared to bi-tempered (22 mistakes) and bi-tempered and focal loss combined (only 16 errors). We believe the improved performance with bi-tempered and focal loss combined is due to the dual challenges present in the dataset: mislabeling and class imbalance. bi-tempered loss effectively addresses the mislabeling issue, while focal loss tackles the class imbalance, collectively enhancing the model's performance. More surprisingly, models trained on the corrupted dataset with the bi-tempered and the focally weighted bi-tempered losses perform better than the models trained on the original dataset. This might be caused by the data corruption showing the effect of a data augmentation step. These preliminary results come from a single train-validation-test split and additional experiments are necessary to confirm these promising findings.

Table 6.7: Results on test images using various loss functions for a training dataset with 10% corruption.

Structures	**Total structures**	**Focal**				**Bi-tempered**				**Bi-tempered + Focal**			
		Dice score	Correct predictions (out of 30)	Mistakes		Dice score	Correct predictions (out of 30)	Mistakes		Dice score	Correct predictions (out of 30)	Mistakes	
				False positive	False negative			False positive	False negative			False positive	False negative
Br1a	30	0.79	29	0	1	0.75	29	0	1	0.80	30	0	0
Br1b	30	0.68	30	0	0	0.80	29	0	1	0.76	30	0	0
Br2a	17	0.57	22	0	8	0.72	27	0	3	0.67	27	1	2
Br2b	17	0.43	15	0	15	0.57	25	0	5	0.47	22	0	8
Cb1	30	0.70	29	0	1	0.66	28	0	2	0.68	29	0	1
Cb2	30	0.68	29	0	1	0.65	28	0	2	0.65	30	0	0
Ch1	30	0.81	30	0	0	0.83	30	0	0	0.81	30	0	0
Ch2	30	0.83	29	0	1	0.81	30	0	0	0.74	30	0	0
Cl1	30	0.49	29	0	1	0.62	27	0	3	0.63	29	0	1
Cl2	30	0.52	30	0	0	0.65	30	0	0	0.60	30	0	0
D1	30	0.73	30	0	0	0.72	30	0	0	0.62	30	0	0
D2	30	0.82	30	0	0	0.74	30	0	0	0.67	30	0	0
En1	30	0.77	30	0	0	0.74	29	0	1	0.75	30	0	0
En2	30	0.78	30	0	0	0.66	30	0	0	0.70	30	0	0
Hm1	30	0.84	30	0	0	0.85	30	0	0	0.82	30	0	0
Hm2	30	0.86	30	0	0	0.84	30	0	0	0.84	30	0	0
M1	28	0.59	26	0	4	0.66	30	0	0	0.55	30	0	0
M2	29	0.53	29	0	1	0.61	30	0	0	0.54	30	0	0
N	30	0.86	30	0	0	0.85	29	0	1	0.84	30	0	0
Oc1	20	0.62	29	1	0	0.60	28	2	0	0.56	29	1	0
Oc2	20	0.61	29	1	0	0.61	29	1	0	0.41	28	2	0
Op1	30	0.90	30	0	0	0.89	30	0	0	0.87	30	0	0
Op2	30	0.89	30	0	0	0.83	30	0	0	0.86	30	0	0
P	30	0.91	30	0	0	0.91	30	0	0	0.88	30	0	0
Total	**671**			**35**				**22**				**16**	

6.6 Conclusions

We have implemented a semantic segmentation-based approach for uncovering the missing, occluded, faint and weak bone structures of zebrafish larvae (9 dpf) from two microscopy image datasets. For the lateral view dataset, we are able to achieve around 98% accuracy (at structure level) in identifying the target bone structures from the test images. In the ventral view dataset, the performance seems more objective as there are overlapping, blur, noisy structures that may lead to subjectivity in the manual annotations. Our approach uses a simple end-to-end deep learning methodology in which the presence or absence of missing structures are automatically reported by the model while simultaneously segmenting faint, small, and overlapping structures. This mitigates the challenges faced by experts during manual visual observation. Preliminary experiments have also shown that the approach can tolerate a number of 'mislabeled/corrupted' annotations if trained with bi-tempered loss function, possibly allowing for the experts to reconsider their previous annotations. We expect that our work will ease the problem of

identifying missing, weak, faint and occluded bone structures in future bone related or morphometric studies even in the presence of mislabeled or corrupted datasets.

CHAPTER 7

CONCLUSIONS AND FUTURE PERSPECTIVES

This thesis provides a thorough exploration of current image analysis methods and advanced machine learning (ML), specifically deep learning (DL) models to address morphometric and phenotype studies in aquaculture and biomedical research.

In Chapter 2, we laid out the key concepts and components of ML and DL, covering supervised learning techniques and their relevance to our research. By delving into optimization strategies, loss functions, CNN architectures and transfer learning methods, we provided a robust foundation for implementing deep learning models that can be optimized for specific bioimage analysis tasks.

By reviewing the existing image analysis tools, algorithms, and methodologies in Chapter 3, we established a foundational understanding of the state-of-the-art techniques available for tasks such as image segmentation, phenotype classification, anatomical landmark detection, and behavior tracking in various fish species. This background informed the development and application of novel automated methods for analyzing fish bioimages, focusing particularly on bone development studies.

Building on these principles and practices and noting the lack of tools for specific tasks of the biomedaqu project partners, we implemented bioimage segmentation methods in Chapter 4, focusing on segmenting the operculum and head regions in red-channel microscopy images of zebrafish larvae. This segmentation allowed for accurate measurement of the operculum-to-head ratio, providing a quantitative metric for studying mineralization in bone development. By addressing class imbalance issues through advanced loss functions and a two-step segmentation process, our end-to-end segmentation

approach offers a significant advancement for automated morphometric analysis.

Chapter 5 extended this work by applying deep learning for anatomical landmark detection across multiple datasets of fish species. We evaluated and compared different regression strategies, demonstrating that heatmap-based regression with an exponential generation function and U-Net architecture yielded the most accurate results across datasets. This approach provides a reliable, scalable solution for bioimage analysis, applicable to both biomedical and aquaculture research settings.

In Chapter 6, we focused on detecting and segmenting weak, faint, overlapping, and missing structures in 2D lateral and ventral bioimages of zebrafish larvae. By employing U-Net variants with single and multi-output masks, we demonstrated that deep learning models effectively segment bone structures, particularly in lateral views. Despite challenges such as blurred boundaries and the subjectivity of manual annotations, our model delivered promising results in identifying missing structures while accurately segmenting faint, blurred, weak, and overlapping structures. Additionally, it exhibited resilience to mislabeled data. This ability to automatically detect missing bone structures and segment weak, blurred, and overlapping ones has the potential to significantly enhance future bone development studies by reducing manual effort and ensuring greater consistency in analysis.

7.1 Future perspectives

This thesis offers a comprehensive foundation in automating bioimage analysis for morphometric and phenotype studies related to fish bone development. The work presented here opens several avenues for future research and practical applications that could further advance bioimage analysis methodologies and their applications in aquaculture and biomedical research. In this section, we outline potential directions and improvements for future studies.

Enhancing model robustness and generalizability: Although our segmentation and landmark detection models demonstrated promising performance across multiple fish species, increasing their robustness on more diverse datasets would enhance their applicability in wider aquaculture and biomedical settings. Future research could integrate domain adaptation techniques [69] more extensively, allowing models trained on specific species or imaging conditions to generalize effectively across new species, imaging modalities, or laboratory setups. Additionally, further exploration of robust, lightweight models for field deployment could enable practical applications in aquaculture monitoring

systems.

Addressing annotation and dataset quality: Given the challenges faced in accurately annotating biomedical images, as discussed in Chapter 6, future studies should explore semi-supervised or self-supervised (unsupervised) learning approaches [80] to alleviate the need for extensive, high-quality annotations. Developing techniques that tolerate or even utilize noisy annotations and corrupted datasets could help improve model performance and reliability, especially when expert-annotated datasets are limited or inconsistently labeled.

Advancements in Multi-task and Multi-output models: The multi-output segmentation methods used for detecting missing structures in the ventral view (Chapter 6) demonstrated the potential of deep learning to perform complex analyses within a single framework. Future studies might expand on this approach by integrating multi-task learning such as discussed in [68], where segmentation, classification, and landmark detection tasks are handled simultaneously. This could result in more efficient processing pipelines and reduce computational costs, making real-time applications more feasible in biomedical and aquaculture environments.

Incorporating advanced data augmentation and synthetic data: To further mitigate the effects of class imbalance and limited data, future work could leverage synthetic data generation techniques, such as Generative Adversarial Networks (GANs) or foundational Models [212], to create additional, diverse training samples. Augmenting training datasets with realistic synthetic images may help address underrepresented classes in bioimage segmentation and anatomical landmark detection, potentially improving model accuracy and resilience to variations in real-world applications.

Exploring transfer learning in greater depth: As discussed in Chapter 2, transfer learning holds promise to reduce training time and resource requirements. Future work could test transfer learning frameworks more extensively, particularly across species with similar morphometric features. Investigating the impact of pre-trained models from related domains or even from other medical imaging tasks could provide useful insights and accelerate the adoption of these tools in new or emerging areas of bioimage analysis. A thorough literature survey of transfer learning in medical imaging is found in [95].

Real-world application and validation in aquaculture: Validating these models in operational aquaculture environments presents a valuable next step. Deployment trials, where these algorithms are tested on real-world aquaculture data [119], would enable practical validation and refinement of the proposed methods. Such trials could lead to adaptive, user-friendly tools for aquaculture practitioners, especially in areas like

automated disease detection, behavioral tracking, and population health monitoring etc.

Multimodal capabilities (e.g., ChatGPT-like models): This thesis explores vision-based deep learning methods, specifically convolutional neural networks (CNNs), developed before the rise of more advanced foundational models like ChatGPT and Vision Transformers (ViTs) [15]. These large-scale models, trained on extensive datasets using unsupervised or self-supervised learning techniques, are designed to be general-purpose and can be refined or adapted for various applications with minimal additional training. Although their multimodal capabilities enable them to handle diverse downstream tasks without requiring training from scratch, these models should be applied carefuly in medical image analysis, considering the specific requirements of each use case. While CNNs have demonstrated strong performance in specialized bioimage analysis tasks, general-purpose multimodal foundational models like ChatGPT are still not yet flexible enough for complex bioimage analysis applications where fine-grained structures have to be detected, such as bioimage segmentation and anatomical landmark detection covered in this thesis.

Using additional collected datasets: Some datasets collected by our collaborators (see Introduction chapter) were not exploited during this thesis. These datasets are openly available on a web-based collaborative platform [124]. Future research may leverage them to further improve AI-driven automation in bioimage analysis, particularly for studies related to bone development in both model and aquaculture fish species.

7.2 Final Remarks

In conclusion, this thesis advances the field of automated bioimage analysis for fish morphometric and phenotype studies by developing adaptable, accurate methods that address key challenges in segmentation, landmark detection, and structure identification. The automated methodologies presented here not only streamline the analytical workflow but are expected to also reduce dependency on labor-intensive manual annotations, which should make bioimage analysis more efficient and scalable. Future studies could extend these methods to other species and imaging conditions, and apply these tools in real-world aquaculture and biomedical research environments. By pursuing the future directions, researchers could enhance the impact of automated bioimage analysis in morphometric studies, making these tools more accessible, accurate, and applicable to a broader range of species and research goals. Ultimately, these developments have the potential to enhance and expand bioimage analysis, contributing to both sustainable aquaculture

practices and advancements in biomedical research.

Bibliography

[1] *cs-230 - deep learning by stanford.edu*.
https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks/.
Accessed on June 10, 2024.

[2] *Zebrafish market report overview*.
https://www.businessresearchinsights.com/market-reports/zebrafish-market-101941/.
Accessed in January, 2025.

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015.
Software available from tensorflow.org.

[4] A. Abu-Siniyeh and W. Al-Zyoud, *Highlights on selected microscopy techniques to study zebrafish developmental biology*, Laboratory Animal Research, 36 (2020), pp. 1–10.

[5] J. Aceto, R. Nourizadeh-Lillabadi, S. Bradamante, J. A. Maier, P. Alestrom, J. J. Van Loon, and M. Muller, *Effects of microgravity simulation on zebrafish transcriptomes and bone physiology—exposure starting at 5 days post fertilization*, npj Microgravity, 2 (2016), pp. 1–8.

[6] J. Aceto, R. Nourizadeh-Lillabadi, R. Maree, N. Dardenne, N. Jeanray, L. Wehenkel, P. Aleström, J. J. van Loon, and M. Muller, *Zebrafish*

bone and general physiology are differently affected by hormones or changes in gravity, PloS one, 10 (2015), p. e0126928.

[7] M. S. AHMED, T. T. AURPA, AND M. A. K. AZAD, *Fish disease detection using image based machine learning technique in aquaculture*, Journal of King Saud University-Computer and Information Sciences, 34 (2022), pp. 5170–5182.

[8] A. A. AKERBERG, C. E. BURNS, C. G. BURNS, AND C. NGUYEN, *Deep learning enables automated volumetric assessments of cardiac function in zebrafish*, Disease models & mechanisms, 12 (2019), p. dmm040188.

[9] S. ALI, D. L. CHAMPAGNE, H. P. SPAINK, AND M. K. RICHARDSON, *Zebrafish embryos and larvae: a new generation of disease models and drug screens*, Birth Defects Research Part C: Embryo Today: Reviews, 93 (2011), pp. 115–133.

[10] E. ALPAYDIN, *Introduction to machine learning*, MIT press, 2020.

[11] E. AMID, M. K. WARMUTH, R. ANIL, AND T. KOREN, *Robust bi-tempered logistic loss based on bregman divergences*, Advances in Neural Information Processing Systems, 32 (2019).

[12] M. ANDRILUKA, L. PISHCHULIN, P. GEHLER, AND B. SCHIELE, *2d human pose estimation: New benchmark and state of the art analysis*, in Proceedings of the IEEE Conference on computer Vision and Pattern Recognition, 2014, pp. 3686–3693.

[13] P. ASHARANI, K. KEUPP, O. SEMLER, W. WANG, Y. LI, H. THIELE, G. YIGIT, E. POHL, J. BECKER, P. FROMMOLT, ET AL., *Attenuated bmp1 function compromises osteogenesis, leading to bone fragility in humans and zebrafish*, The American Journal of Human Genetics, 90 (2012), pp. 661–674.

[14] B. AUBERT, C. VAZQUEZ, T. CRESSON, S. PARENT, AND J. DE GUISE, *Automatic spine and pelvis detection in frontal x-rays using deep neural networks for patch displacement learning*, in 2016 ieee 13th international symposium on biomedical imaging (isbi), IEEE, 2016, pp. 1426–1429.

[15] M. AWAIS, M. NASEER, S. KHAN, R. M. ANWER, H. CHOLAKKAL, M. SHAH, M.-H. YANG, AND F. S. KHAN, *Foundation models defining a new era in vision: a survey and outlook*, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2025).

[16] F. BABAEI, T. L. C. HONG, K. YEUNG, S. H. CHENG, AND Y. W. LAM, *Contrast-enhanced x-ray micro-computed tomography as a versatile method for anatomical studies of adult zebrafish*, Zebrafish, 13 (2016), pp. 310–316.

[17] S. S. BASHA, S. R. DUBEY, V. PULABAIGARI, AND S. MUKHERJEE, *Impact of fully connected layers on performance of convolutional neural networks for image classification*, Neurocomputing, 378 (2020), pp. 112–119.

[18] B. BAUER, A. MALLY, AND D. LIEDTKE, *Zebrafish embryos and larvae as alternative animal models for toxicity testing*, International journal of molecular sciences, 22 (2021), p. 13417.

[19] M. C. BECKMANN, J. F. GILLIAM, AND R. B. LANGERHANS, *X-ray imaging as a time-saving, non-invasive technique for diet analysis*, Fisheries research, 161 (2015), pp. 1–7.

[20] M. BENNET, A. AKIVA, D. FAIVRE, G. MALKINSON, K. YANIV, S. ABDELILAH-SEYFRIED, P. FRATZL, AND A. MASIC, *Simultaneous raman microspectroscopy and fluorescence imaging of bone mineralization in living zebrafish larvae*, Biophysical journal, 106 (2014), pp. L17–L19.

[21] J. BERTELS, T. EELBODE, M. BERMAN, D. VANDERMEULEN, F. MAES, R. BISSCHOPS, AND M. B. BLASCHKO, *Optimizing the dice score and jaccard index for medical image segmentation: Theory and practice*, in Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part II 22, Springer, 2019, pp. 92–100.

[22] G. BIAU, *Analysis of a random forests model*, The Journal of Machine Learning Research, 13 (2012), pp. 1063–1095.

[23] E. BISONG AND E. BISONG, *Google automl: cloud vision*, Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners, (2019), pp. 581–598.

[24] F. L. BOOKSTEIN, *Combining the tools of geometric morphometrics*, in Advances in morphometrics, Springer, 1996, pp. 131–151.

[25] T. BRAUNBECK, B. KAIS, E. LAMMER, J. OTTE, K. SCHNEIDER, D. STENGEL, AND R. STRECKER, *The fish embryo test (fet): origin, applications, and future*, Environmental Science and Pollution Research, 22 (2015), pp. 16247–16261.

[26] B. BRUNEEL AND P. E. WITTEN, *Power and challenges of using zebrafish as a model for skeletal tissue imaging*, Connective tissue research, 56 (2015), pp. 161–173.

[27] S. M. BUGEL AND R. L. TANGUAY, *Multidimensional chemobehavior analysis of flavonoids and neuroactive compounds in zebrafish*, Toxicology and applied pharmacology, 344 (2018), pp. 23–34.

[28] C. CAETANO DA SILVA, A. OSTERTAG, R. RAMAN, M. MULLER, M. COHEN-SOLAL, AND C. COLLET, *wnt11f2 zebrafish, an animal model for development and new insights in bone formation*, Zebrafish, 20 (2023), pp. 1–9.

[29] S. CAI, Y. TIAN, H. LUI, H. ZENG, Y. WU, AND G. CHEN, *Dense-unet: a novel multiphoton in vivo cellular image segmentation model based on a convolutional neural network*, Quantitative imaging in medicine and surgery, 10 (2020), p. 1275.

[30] D. ČAPEK, M. SAFROSHKIN, H. MORALES-NAVARRETE, N. TOULANY, G. ARUTYUNOV, A. KURZBACH, J. BIHLER, J. HAGAUER, S. KICK, F. JONES, ET AL., *Embryonet: using deep learning to link embryonic phenotypes to signaling pathways*, Nature Methods, (2023), pp. 1–9.

[31] A. CARLETTI, C. CARDOSO, J. LOBO-ARTEAGA, S. SALES, D. JULIAO, I. FERREIRA, P. CHAINHO, M. A. DIONÍSIO, M. J. GAUDÊNCIO, C. AFONSO, ET AL., *Antioxidant and anti-inflammatory extracts from sea cucumbers and tunicates induce a pro-osteogenic effect in zebrafish larvae*, Frontiers in Nutrition, 9 (2022), p. 888360.

[32] M. CARNOVALI, G. BANFI, M. MARIOTTI, ET AL., *Zebrafish models of human skeletal disorders: embryo and adult swimming together*, BioMed Research International, 2019 (2019).

[33] M. J. CARREIRA, N. VILA-BLANCO, P. CABEZAS-SAINZ, AND L. SÁNCHEZ, *Zftool: A software for automatic quantification of cancer cell mass evolution in zebrafish*, Applied Sciences, 11 (2021), p. 7721.

[34] S. CASSAR, I. ADATTO, J. L. FREEMAN, J. T. GAMSE, I. ITURRIA, C. LAWRENCE, A. MURIANA, R. T. PETERSON, S. VAN CRUCHTEN, AND L. I. ZON, *Use of zebrafish in drug discovery toxicology*, Chemical research in toxicology, 33 (2019), pp. 95–118.

[35] C. CHAKRABORTY, C. H. HSU, Z. H. WEN, C. S. LIN, AND G. AGORAMOORTHY, *Zebrafish: a complete animal model for in vivo drug discovery and development*, Current drug metabolism, 10 (2009), pp. 116–124.

[36] O. CHAPELLE, B. SCHOLKOPF, AND A. ZIEN, *Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]*, IEEE Transactions on Neural Networks, 20 (2009), pp. 542–542.

[37] R.-C. CHEN, C. DEWI, S.-W. HUANG, AND R. E. CARAKA, *Selecting critical features for data classification based on machine learning methods*, Journal of Big Data, 7 (2020), p. 52.

[38] T. CHEN, T. HE, M. BENESTY, V. KHOTILOVICH, Y. TANG, H. CHO, K. CHEN, R. MITCHELL, I. CANO, T. ZHOU, ET AL., *Xgboost: extreme gradient boosting*, R package version 0.4-2, 1 (2015), pp. 1–4.

[39] T. CHEN, M. LI, Y. LI, M. LIN, N. WANG, M. WANG, T. XIAO, B. XU, C. ZHANG, AND Z. ZHANG, *Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems*, arXiv preprint arXiv:1512.01274, (2015).

[40] X. CHEN, Z. WANG, N. DUAN, G. ZHU, E. M. SCHWARZ, AND C. XIE, *Osteoblast–osteoclast interactions*, Connective tissue research, 59 (2018), pp. 99–107.

[41] T.-Y. CHOI, T.-I. CHOI, Y.-R. LEE, S.-K. CHOE, AND C.-H. KIM, *Zebrafish as an animal model for biomedical research*, Experimental & Molecular Medicine, 53 (2021), pp. 310–317.

[42] F. CHOLLET ET AL., *Keras*, 2015.

[43] S. COTTI, A. HUYSSEUNE, D. LARIONOVA, W. KOPPE, A. FORLINO, AND P. E. WITTEN, *Compression fractures and partial phenotype rescue with a low phosphorus diet in the chihuahua zebrafish osteogenesis imperfecta model*, Frontiers in Endocrinology, 13 (2022), p. 851879.

[44] C. C. CUBBAGE AND P. M. MABEE, *Development of the cranium and paired fins in the zebrafish danio rerio (ostariophysi, cyprinidae)*, Journal of Morphology, 229 (1996), pp. 121–160.

[45] K. M. DA SILVA, E. ITURROSPE, C. BARS, D. KNAPEN, S. VAN CRUCHTEN, A. COVACI, AND A. L. VAN NUIJS, *Mass spectrometry-based zebrafish toxicometabolomics: a review of analytical and data quality challenges*, Metabolites, 11 (2021), p. 635.

[46] N. S. DAMANHURI, M. F. M. ZAMRI, N. A. OTHMAN, S. A. SHAMSUDDIN, B. C. C. MENG, M. H. ABBAS, AND A. AHMAD, *An automated length measurement system for tilapia fish based on image processing technique*, in IOP Conference Series: Materials Science and Engineering, vol. 1088, IOP Publishing, 2021, p. 012049.

[47] F. DE CHAUMONT, S. DALLONGEVILLE, N. CHENOUARD, N. HERVÉ, S. POP, T. PROVOOST, V. MEAS-YEDID, P. PANKAJAKSHAN, T. LECOMTE, Y. LE MONTAGNER, ET AL., *Icy: an open bioimage informatics platform for extended reproducible research*, Nature methods, 9 (2012), pp. 690–696.

[48] Z. DELLACQUA, C. DI BIAGIO, C. COSTA, P. POUSÃO-FERREIRA, L. RIBEIRO, M. BARATA, P. J. GAVAIA, F. MATTEI, A. FABRIS, M. IZQUIERDO, ET AL., *Distinguishing the effects of water volumes versus stocking densities on the skeletal quality during the pre-ongrowing phase of gilthead seabream (sparus aurata)*, Animals, 13 (2023), p. 557.

[49] J. DENG, W. DONG, R. SOCHER, L.-J. LI, K. LI, AND L. FEI-FEI, *Imagenet: A large-scale hierarchical image database*, in 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[50] Y. DING, D. J. VANSELOW, M. A. YAKOVLEV, S. R. KATZ, A. Y. LIN, D. P. CLARK, P. VARGAS, X. XIN, J. E. COPPER, V. A. CANFIELD, ET AL., *Computational 3d histological phenotyping of whole zebrafish by x-ray histotomography*, Elife, 8 (2019), p. e44898.

[51] B. DONG, L. SHAO, M. DA COSTA, O. BANDMANN, AND A. F. FRANGI, *Deep learning for automatic cell detection in wide-field microscopy zebrafish images*, in 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), IEEE, 2015, pp. 772–776.

[52] W. DRIEVER, L. SOLNICA-KREZEL, A. SCHIER, S. NEUHAUSS, J. MALICKI, D. STEMPLE, D. STAINIER, F. ZWARTKRUIS, S. ABDELILAH, Z. RANGINI, ET AL., *A genetic screen for mutations affecting embryogenesis in zebrafish*, Development, 123 (1996), pp. 37–46.

[53] O. DÜRR AND B. SICK, *Single-cell phenotype classification using deep convolutional neural networks*, Journal of biomolecular screening, 21 (2016), pp. 998–1003.

[54] B. F. EAMES, A. DELAURIER, B. ULLMANN, T. R. HUYCKE, J. T. NICHOLS, J. DOWD, M. MCFADDEN, M. M. SASAKI, AND C. B. KIMMEL, *Fishface: interactive atlas of zebrafish craniofacial development at cellular resolution*, BMC developmental biology, 13 (2013), pp. 1–11.

[55] C. A. EDWARDS, A. GOYAL, A. E. RUSHEEN, A. Z. KOUZANI, AND K. H. LEE, *Deepnavnet: Automated landmark localization for neuronavigation*, Frontiers in Neuroscience, 15 (2021), p. 730.

[56] J. G. EVANS AND P. MATSUDAIRA, *Linking microscopy and high content screening in large-scale biomedical research*, High Content Screening, (2007), pp. 33–38.

[57] I. A. FIEDLER, F. N. SCHMIDT, E. M. WÖLFEL, C. PLUMEYER, P. MILOVANOVIC, R. GIOIA, F. TONELLI, H. A. BALE, K. JÄHN, R. BESIO, ET AL., *Severely impaired bone material quality in chihuahua zebrafish resembles classical dominant human osteogenesis imperfecta*, Journal of Bone and Mineral Research, 33 (2018), pp. 1489–1499.

[58] S. FRAGKOULIS, A. PRINTZI, G. GELADAKIS, N. KATRIBOUZAS, AND G. KOUMOUNDOUROS, *Recovery of haemal lordosis in gilthead seabream (sparus aurata l.)*, Scientific reports, 9 (2019), pp. 1–11.

[59] Y. FREUND AND R. E. SCHAPIRE, *A decision-theoretic generalization of on-line learning and an application to boosting*, Journal of computer and system sciences, 55 (1997), pp. 119–139.

[60] J. H. FRIEDMAN, *Stochastic gradient boosting*, Computational statistics & data analysis, 38 (2002), pp. 367–378.

[61] D. GENEST, É. PUYBAREAU, M. LÉONARD, J. COUSTY, N. DE CROZÉ, AND H. TALBOT, *High throughput automated detection of axial malformations in medaka embryo*, Computers in biology and medicine, 105 (2019), pp. 157–168.

[62] P. GEURTS, D. ERNST, AND L. WEHENKEL, *Extremely randomized trees*, Machine learning, 63 (2006), pp. 3–42.

[63] T. GJEDREM, N. ROBINSON, ET AL., *Advances by selective breeding for aquatic species: a review*, Agricultural Sciences, 5 (2014), p. 1152.

[64] X. GLOROT AND Y. BENGIO, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[65] R. C. GONZALEZ, *Digital image processing*, Pearson education india, 2009.

[66] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep learning*, MIT press, 2016.

[67] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative adversarial networks*, Communications of the ACM, 63 (2020), pp. 139–144.

[68] S. GRAHAM, Q. D. VU, M. JAHANIFAR, S. E. A. RAZA, F. MINHAS, D. SNEAD, AND N. RAJPOOT, *One model is all you need: multi-task learning enables simultaneous histology image segmentation and classification*, Medical Image Analysis, 83 (2023), p. 102685.

[69] R. GU, G. WANG, J. LU, J. ZHANG, W. LEI, Y. CHEN, W. LIAO, S. ZHANG, K. LI, D. N. METAXAS, ET AL., *Cddsa: Contrastive domain disentanglement and style augmentation for generalizable medical image segmentation*, Medical Image Analysis, 89 (2023), p. 102904.

[70] H. GUAN AND M. LIU, *Domain adaptation for medical image analysis: A survey*, IEEE Transactions on Biomedical Engineering, 69 (2022), pp. 1173–1185.

[71] Y. GUO, L. ZHANG, Y. HU, X. HE, AND J. GAO, *Ms-celeb-1m: A dataset and benchmark for large-scale face recognition*, in European conference on computer vision, Springer, 2016, pp. 87–102.

[72] T. HASTIE, R. TIBSHIRANI, J. H. FRIEDMAN, AND J. H. FRIEDMAN, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2, Springer, 2009.

[73] K. He, X. Zhang, S. Ren, and J. Sun, *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.

[74] ——, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[75] A. J. Hill, H. Teraoka, W. Heideman, and R. E. Peterson, *Zebrafish as a model vertebrate for investigating chemical toxicity*, Toxicological sciences, 86 (2005), pp. 6–19.

[76] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al., *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*, 2001.

[77] H. Høgset, C. C. Horgan, J. P. Armstrong, M. S. Bergholt, V. Torraca, Q. Chen, T. J. Keane, L. Bugeon, M. J. Dallman, S. Mostowy, et al., *In vivo biomolecular imaging of zebrafish embryos using confocal raman spectroscopy*, Nature Communications, 11 (2020), p. 6172.

[78] W.-C. Hu, L.-B. Chen, B.-K. Huang, and H.-M. Lin, *A computer vision-based intelligent fish feeding system using deep learning techniques for aquaculture*, IEEE Sensors Journal, 22 (2022), pp. 7185–7194.

[79] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, *Unet 3+: A full-scale connected unet for medical image segmentation*, in ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, 2020, pp. 1055–1059.

[80] S.-C. Huang, A. Pareek, M. Jensen, M. P. Lungren, S. Yeung, and A. S. Chaudhari, *Self-supervised learning for medical image classification: a systematic review and implementation guidelines*, NPJ Digital Medicine, 6 (2023), p. 74.

[81] W. Huang, C. Yang, and T. Hou, *Spine landmark localization with combining of heatmap regression and direct coordinate regression*, arXiv preprint arXiv:2007.05355, (2020).

[82] S. Hussain, R. Aponte-Rivera, R. M. Barghout, J. G. Trapani, and K. S. Kindt, *In vivo analysis of hair cell sensory organs in zebrafish: From morphol-*

ogy to function, Developmental, physiological, and functional neurobiology of the inner ear, (2022), pp. 175–220.

[83] N.-N. HUYNH, M. JUN, H. T. P. NGUYEN, C. SHIN, AND H. JEONG, *Preliminary study on fish tracking in indoor aquaculture through deep learning*.

[84] B. IBRAGIMOV AND T. VRTOVEC, *Landmark-based statistical shape representations*, in Statistical Shape and Deformation Analysis, Elsevier, 2017, pp. 89–113.

[85] E. INSAFUTDINOV, L. PISHCHULIN, B. ANDRES, M. ANDRILUKA, AND B. SCHIELE, *Deepercut: A deeper, stronger, and faster multi-person pose estimation model*, in European conference on computer vision, Springer, 2016, pp. 34–50.

[86] S. IOFFE AND C. SZEGEDY, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in International conference on machine learning, pmlr, 2015, pp. 448–456.

[87] O. ISHAQ, J. NEGRI, M.-A. BRAY, A. PACUREANU, R. T. PETERSON, AND C. WÄHLBY, *Automated quantification of zebrafish tail deformation for high-throughput drug screening*, in 2013 IEEE 10th International Symposium on Biomedical Imaging, IEEE, 2013, pp. 902–905.

[88] O. ISHAQ, S. K. SADANANDAN, AND C. WÄHLBY, *Deep fish: deep learning–based classification of zebrafish deformation for high-throughput screening*, SLAS Discovery: Advancing Life Sciences R&D, 22 (2017), pp. 102–107.

[89] M. IVERSEN, B. FINSTAD, R. S. MCKINLEY, AND R. A. ELIASSEN, *The efficacy of metomidate, clove oil, aqui-s™ and benzoak® as anaesthetics in atlantic salmon (salmo salar l.) smolts, and their potential stress-reducing capacity*, Aquaculture, 221 (2003), pp. 549–566.

[90] J. K. JAISWAL AND R. SAMIKANNU, *Application of random forest algorithm on feature subset selection and classification and regression*, in 2017 world congress on computing and communication technologies (WCCCT), Ieee, 2017, pp. 65–68.

[91] S. JARQUE, M. RUBIO-BROTONS, J. IBARRA, V. ORDOÑEZ, S. DYBALLA, R. MIÑANA, AND J. TERRIENTE, *Morphometric analysis of developing zebrafish*

embryos allows predicting teratogenicity modes of action in higher vertebrates, Reproductive Toxicology, 96 (2020), pp. 337–348.

[92] N. JEANRAY, R. MARÉE, B. PRUVOT, O. STERN, P. GEURTS, L. WEHENKEL, AND M. MULLER, *Phenotype classification of zebrafish embryos by supervised learning*, PloS one, 10 (2015), p. e0116989.

[93] R. A. JONES, M. J. RENSHAW, AND D. J. BARRY, *Automated staging of zebrafish embryos with deep learning*, Life Science Alliance, 7 (2024).

[94] K. KHABARLAK AND L. KORIASHKINA, *Fast facial landmark detection and applications: A survey*, arXiv preprint arXiv:2101.10808, (2021).

[95] H. E. KIM, A. COSA-LINAN, N. SANTHANAM, M. JANNESARI, M. E. MAROS, AND T. GANSLANDT, *Transfer learning for medical image classification: a literature review*, BMC medical imaging, 22 (2022), p. 69.

[96] C. B. KIMMEL, W. W. BALLARD, S. R. KIMMEL, B. ULLMANN, AND T. F. SCHILLING, *Stages of embryonic development of the zebrafish*, Developmental dynamics, 203 (1995), pp. 253–310.

[97] C. B. KIMMEL, A. DELAURIER, B. ULLMANN, J. DOWD, AND M. MCFADDEN, *Modes of developmental outgrowth and shaping of a craniofacial bone in zebrafish*, PloS one, 5 (2010), p. e9475.

[98] D. P. KINGMA AND J. BA, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).

[99] U. KORNAK AND S. MUNDLOS, *Genetic disorders of the skeleton: a developmental approach*, The American Journal of Human Genetics, 73 (2003), pp. 447–474.

[100] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *Imagenet classification with deep convolutional neural networks*, Advances in neural information processing systems, 25 (2012).

[101] A. KUCHMIY, G. EFIMOV, AND S. NEDOSPASOV, *Methods for in vivo molecular imaging*, Biochemistry (Moscow), 77 (2012), pp. 1339–1353.

[102] H. W. KUHN, *The hungarian method for the assignment problem*, Naval Research Logistics (NRL), 52 (2005), pp. 7–21.

[103] N. KUMAR, A. CARLETTI, P. J. GAVAIA, M. MULLER, M. L. CANCELA, P. GEURTS, AND R. MARÉE, *Deep learning approaches for head and operculum segmentation in zebrafish microscopy images*, in Computer Analysis of Images and Patterns: 19th International Conference, CAIP 2021, Virtual Event, September 28–30, 2021, Proceedings, Part I 19, Springer, 2021, pp. 154–164.

[104] H. LAI, S. XIAO, Y. PAN, Z. CUI, J. FENG, C. XU, J. YIN, AND S. YAN, *Deep recurrent regression for facial landmark detection*, IEEE Transactions on Circuits and Systems for Video Technology, 28 (2016), pp. 1144–1157.

[105] H. LEE, M. PARK, AND J. KIM, *Cephalometric landmark detection in dental x-ray images using convolutional neural networks*, in Medical imaging 2017: Computer-aided diagnosis, vol. 10134, International Society for Optics and Photonics, 2017, p. 101341W.

[106] S. LEE, J. KIM, J.-Y. BAEK, M.-W. HAN, S. KIM, AND T.-S. CHON, *Pattern analysis of movement behavior of medaka (oryzias latipes): a decision tree approach*, in Computer Analysis of Images and Patterns: 11th International Conference, CAIP 2005, Versailles, France, September 5-8, 2005. Proceedings 11, Springer, 2005, pp. 546–553.

[107] S.-H. LEE, C.-C. KIM, S.-J. KOH, L.-S. SHIN, J.-K. CHO, AND K.-H. HAN, *Egg development and morphology of larva and juvenile of the oryzias latipes*, Development & Reproduction, 18 (2014), p. 173.

[108] C. A. LESSMAN, *The developing zebrafish (danio rerio): A vertebrate model for high-throughput screening of chemical libraries*, Birth Defects Research Part C: Embryo Today: Reviews, 93 (2011), pp. 268–280.

[109] X. LI, H. CHEN, X. QI, Q. DOU, C.-W. FU, AND P.-A. HENG, *H-denseunet: hybrid densely connected unet for liver and tumor segmentation from ct volumes*, IEEE transactions on medical imaging, 37 (2018), pp. 2663–2674.

[110] G. J. LIESCHKE AND P. D. CURRIE, *Animal models of human disease: zebrafish swim into view*, Nature Reviews Genetics, 8 (2007), pp. 353–367.

[111] T.-Y. LIN, P. GOYAL, R. GIRSHICK, K. HE, AND P. DOLLÁR, *Focal loss for dense object detection*, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.

[112] C. LINDNER AND T. F. COOTES, *Fully automatic cephalometric evaluation using random forest regression-voting*, in IEEE International Symposium on Biomedical Imaging (ISBI) 2015–Grand Challenges in Dental X-ray Image Analysis–Automated Detection and Analysis for Diagnosis in Cephalometric X-ray Image, 2015.

[113] X. LIU, K. GAO, B. LIU, C. PAN, K. LIANG, L. YAN, J. MA, F. HE, S. ZHANG, S. PAN, ET AL., *Advances in deep learning-based medical image analysis*, Health Data Science, (2021).

[114] Y. LIU, P. MA, P. A. CASSIDY, R. CARMER, G. ZHANG, P. VENKATRAMAN, S. A. BROWN, C. P. PANG, W. ZHONG, M. ZHANG, ET AL., *Statistical analysis of zebrafish locomotor behaviour by generalized linear mixed models*, Scientific reports, 7 (2017), pp. 1–9.

[115] I. LLORENTE, J. FERNÁNDEZ-POLANCO, E. BARAIBAR-DIEZ, M. D. ODRIOZOLA, T. BJØRNDAL, F. ASCHE, J. GUILLEN, L. AVDELAS, R. NIELSEN, M. COZZOLINO, ET AL., *Assessment of the economic performance of the seabream and seabass aquaculture industry in the european union*, Marine Policy, 117 (2020), p. 103876.

[116] J. LONG, E. SHELHAMER, AND T. DARRELL, *Fully convolutional networks for semantic segmentation*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431–3440.

[117] D. R. LOVE, F. B. PICHLER, A. DODD, B. R. COPP, AND D. R. GREENWOOD, *Technology for high-throughput screens: the present and future using zebrafish*, Current opinion in biotechnology, 15 (2004), pp. 564–571.

[118] B. A. LOY, C. BOGLIONE, AND S. CATAUDELLA, *Geometric morphometrics and morpho-anatomy: a combined tool in the study of sea bream (sparus aurata, sparidae) shape*, Journal of Applied Ichthyology, 15 (1999), pp. 104–110.

[119] C. G. LUTZ, *The rise of ai in aquaculture*. https://thefishsite.com/articles/the-rise-of-ai-in-aquaculture-artificial-intelligence.
Accessed in February, 2025.

[120] C. A. MACRAE AND R. T. PETERSON, *Zebrafish as tools for drug discovery*, Nature reviews Drug discovery, 14 (2015), pp. 721–731.

[121] S. MAHPOD, R. DAS, E. MAIORANA, Y. KELLER, AND P. CAMPISI, *Facial landmarks localization using cascaded neural networks*, Computer Vision and Image Understanding, 205 (2021), p. 103171.

[122] V. MAINI AND S. SABRI, *Machine learning for humans*, Online: https://medium.com/machine-learning-for-humans, (2017).

[123] S. MALIK, T. KUMAR, AND A. SAHOO, *A novel approach to fish disease diagnostic system based on machine learning*, Adv. Image Video Process, 5 (2017), pp. 49–57.

[124] R. MARÉE, L. ROLLUS, B. STÉVENS, R. HOYOUX, G. LOUPPE, R. VANDAELE, J.-M. BEGON, P. KAINZ, P. GEURTS, AND L. WEHENKEL, *Collaborative analysis of multi-gigapixel imaging data using cytomine*, Bioinformatics, 32 (2016), pp. 1395–1401.

[125] D. MARRABLE, S. TIPPAYA, K. BARKER, E. HARVEY, S. L. BIERWAGEN, M. WYATT, S. BAINBRIDGE, AND M. STOWAR, *Generalised deep learning model for semi-automated length measurement of fish in stereo-bruvs*, Frontiers in Marine Science, 10 (2023), p. 1171625.

[126] J. A. MARTOS-SITCHA, J. M. MANCERA, P. PRUNET, AND L. J. MAGNONI, *Welfare and stressors in fish: Challenges facing aquaculture*, 2020.

[127] A. MATHIS, P. MAMIDANNA, K. M. CURY, T. ABE, V. N. MURTHY, M. W. MATHIS, AND M. BETHGE, *Deeplabcut: markerless pose estimation of user-defined body parts with deep learning*, Nature neuroscience, 21 (2018), pp. 1281–1289.

[128] G. D. MERRIFIELD, J. MULLIN, L. GALLAGHER, C. TUCKER, M. A. JANSEN, M. DENVIR, AND W. M. HOLMES, *Rapid and recoverable in vivo magnetic resonance imaging of the adult zebrafish at 7t*, Magnetic Resonance Imaging, 37 (2017), pp. 9–15.

[129] R. MIKUT, A. BARTSCHAT, W. DONEIT, J. Á. G. ORDIANO, B. SCHOTT, J. STEGMAIER, S. WACZOWICZ, AND M. REISCHL, *The matlab toolbox scixminer: User's manual and programmer's guide*, arXiv preprint arXiv:1704.03298, (2017).

[130] R. MIKUT, T. DICKMEIS, W. DRIEVER, P. GEURTS, F. A. HAMPRECHT, B. X. KAUSLER, M. J. LEDESMA-CARBAYO, R. MARÉE, K. MIKULA, P. PANTAZIS,

ET AL., *Automated processing of zebrafish imaging data: a survey*, Zebrafish, 10 (2013), pp. 401–421.

[131] O. MIRAT, J. R. STERNBERG, K. E. SEVERI, AND C. WYART, *Zebrazoom: an automated program for high-throughput behavioral analysis and categorization*, Frontiers in neural circuits, 7 (2013), p. 107.

[132] T. M. MITCHELL, *Machine learning*, 1997.

[133] H. MOHSENI AND S. KASAEI, *Automatic localization of cephalometric landmarks*, in 2007 IEEE International Symposium on Signal Processing and Information Technology, IEEE, 2007, pp. 396–401.

[134] P. MORRIS, *Biomedical imaging: applications and advances*, (2014).

[135] D. MYERS-TURNBULL, J. C. TAYLOR, C. HELSELL, T. A. TUMMINO, M. N. MCCARROLL, R. ALEXANDER, C. S. KI, L. GENDELEV, AND D. KOKEL, *Simultaneous classification of neuroactive compounds in zebrafish*, bioRxiv, (2020).

[136] T. NAERT, Ö. ÇIÇEK, P. OGAR, M. BÜRGI, N.-I. SHAIDANI, M. M. KAMINSKI, Y. XU, K. GRAND, M. VUJANOVIC, D. PRATA, ET AL., *Deep learning is widely applicable to phenotyping embryonic development and disease*, Development, 148 (2021), p. dev199664.

[137] A. NAVARRO, I. LEE-MONTERO, D. SANTANA, P. HENRÍQUEZ, M. A. FERRER, A. MORALES, M. SOULA, R. BADILLA, D. NEGRÍN-BÁEZ, M. J. ZAMORANO, ET AL., *Imafish_ml: A fully-automated image analysis software for assessing fish morphometric traits on gilthead seabream (sparus aurata l.), meagre (argyrosomus regius) and red porgy (pagrus pagrus)*, Computers and Electronics in Agriculture, 121 (2016), pp. 66–73.

[138] W. M. NEWMAN AND R. F. SPROULL, *Principles of interactive computer graphics*, McGraw-Hill, Inc., 1979.

[139] I. PADUANO, A. MILETO, AND E. LOFRANO, *A perspective on ai-based image analysis and utilization technologies in building engineering: Recent developments and new directions*, Buildings, 13 (2023), p. 1198.

[140] S. J. PAN AND Q. YANG, *A survey on transfer learning*, IEEE Transactions on knowledge and data engineering, 22 (2009), pp. 1345–1359.

[141] A. PANNU, *Artificial intelligence and its application in different areas*, Artificial Intelligence, 4 (2015), pp. 79–84.

[142] J.-H. PARK, H.-W. HWANG, J.-H. MOON, Y. YU, H. KIM, S.-B. HER, G. SRINIVASAN, M. N. A. ALJANABI, R. E. DONATELLI, AND S.-J. LEE, *Automated identification of cephalometric landmarks: Part 1—comparisons between the latest deep-learning methods yolov3 and ssd*, The Angle Orthodontist, 89 (2019), pp. 903–909.

[143] C. PAYER, D. ŠTERN, H. BISCHOF, AND M. URSCHLER, *Regressing heatmaps for multiple landmark localization using cnns*, in International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2016, pp. 230–238.

[144] C. PAYER, D. STERN, H. BISCHOF, AND M. URSCHLER, *Integrating spatial configuration into heatmap regression based cnns for landmark localization*, Medical image analysis, 54 (2019), pp. 207–219.

[145] R. T. PETERSON AND C. A. MACRAE, *Systematic approaches to toxicology in the zebrafish*, Annual review of pharmacology and toxicology, 52 (2012), pp. 433–453.

[146] B. PRIJONO, *indoml*.
https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/.
Accessed on June 15, 2024.

[147] S. J. PRINCE, *Understanding Deep Learning*, The MIT Press, 2023.

[148] B. PRUVOT, Y. CURÉ, J. DJIOTSA, A. VONCKEN, AND M. MULLER, *Developmental defects in zebrafish for classification of egf pathway inhibitors*, Toxicology and applied pharmacology, 274 (2014), pp. 339–349.

[149] P. RAGHAVENDRA AND T. PULLAIAH, *Biomedical imaging role in cellular and molecular diagnostics*, Adv. Cell Mol. Diagnostics, (2018), pp. 85–111.

[150] R. RAMAN, M. ANTONY, R. NIVELLE, A. LAVERGNE, J. ZAPPIA, G. GUERRERO-LIMÓN, C. CAETANO DA SILVA, P. KUMARI, J. M. SOJAN, C. DEGUELDRE, ET AL., *The osteoblast transcriptome in developing zebrafish reveals key roles*

for extracellular matrix proteins col10a1a and fbln1 in skeletal development and homeostasis, Biomolecules, 14 (2024), p. 139.

[151] S. S. RAUTARAY AND A. AGRAWAL, *Vision based hand gesture recognition for human computer interaction: a survey*, Artificial intelligence review, 43 (2015), pp. 1–54.

[152] J. REDMON, S. DIVVALA, R. GIRSHICK, AND A. FARHADI, *You only look once: Unified, real-time object detection*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[153] M. REISCHL, A. BARTSCHAT, U. LIEBEL, J. GEHRIG, F. MÜLLER, AND R. MIKUT, *Zebrafishminer: an open source software for interactive evaluation of domain-specific fluorescence in zebrafish*, Current Directions in Biomedical Engineering, 3 (2017), pp. 199–202.

[154] G. RIEGLER, M. URSCHLER, M. RUTHER, H. BISCHOF, AND D. STERN, *Anatomical landmark detection in medical applications driven by synthetic data*, in Proceedings of the IEEE international conference on computer vision workshops, 2015, pp. 12–16.

[155] S. J. RIGATTI, *Random forest*, Journal of Insurance Medicine, 47 (2017), pp. 31–39.

[156] J. RIHEL, D. A. PROBER, A. ARVANITES, K. LAM, S. ZIMMERMAN, S. JANG, S. J. HAGGARTY, D. KOKEL, L. L. RUBIN, R. T. PETERSON, ET AL., *Zebrafish behavioral profiling links drugs to biological targets and rest/wake regulation*, Science, 327 (2010), pp. 348–351.

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

[158] D. E. RUMELHART, G. E. HINTON, AND R. J. WILLIAMS, *Learning representations by back-propagating errors*, nature, 323 (1986), pp. 533–536.

[159] S. J. RUSSELL AND P. NORVIG, *Artificial intelligence a modern approach*, London, 2010.

[160] A. H. SABIR AND T. COLE, *The evolving therapeutic landscape of genetic skeletal disorders*, Orphanet Journal of Rare Diseases, 14 (2019), pp. 1–20.

[161] S. S. M. SALEHI, D. ERDOGMUS, AND A. GHOLIPOUR, *Tversky loss function for image segmentation using 3d fully convolutional deep networks*, in International workshop on machine learning in medical imaging, Springer, 2017, pp. 379–387.

[162] N. SAMET AND E. AKBAS, *Hprnet: Hierarchical point regression for whole-body human pose estimation*, Image and Vision Computing, 115 (2021), p. 104285.

[163] S. SANTURKAR, D. TSIPRAS, A. ILYAS, AND A. MADRY, *How does batch normalization help optimization?*, Advances in neural information processing systems, 31 (2018).

[164] R. SAWAKI, D. SATO, H. NAKAYAMA, Y. NAKAGAWA, AND Y. SHIMADA, *Zf-automl: An easy machine-learning-based method to detect anomalies in fluorescent-labelled zebrafish*, Inventions, 4 (2019), p. 72.

[165] S. SCHOLZ, S. FISCHER, U. GÜNDEL, E. KÜSTER, T. LUCKENBACH, AND D. VOELKER, *The zebrafish embryo model in environmental risk assessment—applications beyond acute toxicity testing*, Environmental science and pollution research, 15 (2008), pp. 394–404.

[166] I. W. SELDERSLAGHS, J. HOOYBERGHS, R. BLUST, AND H. E. WITTERS, *Assessment of the developmental neurotoxicity of compounds by measuring locomotor activity in zebrafish embryos and larvae*, Neurotoxicology and teratology, 37 (2013), pp. 44–56.

[167] N. SENGAR, M. K. DUTTA, AND B. SARKAR, *Computer vision based technique for identification of fish quality after pesticide exposure*, International journal of food properties, 20 (2017), pp. 2192–2206.

[168] L. SHAMIR, J. D. DELANEY, N. ORLOV, D. M. ECKLEY, AND I. G. GOLDBERG, *Pattern recognition software and techniques for biological image analysis*, PLoS computational biology, 6 (2010), p. e1000974.

[169] S. SHANG, S. LIN, AND F. CONG, *Zebrafish larvae phenotype classification from bright-field microscopic images using a two-tier deep-learning pipeline*, Applied Sciences, 10 (2020), p. 1247.

[170] S. SHANG, L. LONG, S. LIN, AND F. CONG, *Automatic zebrafish egg phenotype recognition from bright-field microscopic images using deep convolutional neural network*, Applied Sciences, 9 (2019), p. 3362.

[171] K. SIMONYAN AND A. ZISSERMAN, *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556, (2014).

[172] Y. SONG, X. QIAO, Y. IWAMOTO, AND Y.-W. CHEN, *Automatic cephalometric landmark detection on x-ray images using a deep-learning method*, Applied Sciences, 10 (2020), p. 2547.

[173] A. SONNENSCHEIN, D. VANDERZEE, W. R. PITCHERS, S. CHARI, AND I. DWORKIN, *An image database of drosophila melanogaster wings for phenomic and biometric analysis*, GigaScience, 4 (2015), pp. s13742–015.

[174] J. C. SPALL, *Introduction to stochastic search and optimization: estimation, simulation, and control*, John Wiley & Sons, 2005.

[175] K. M. SPOORENDONK, C. L. HAMMOND, L. F. HUITEMA, J. VANOEVELEN, AND S. SCHULTE-MERKER, *Zebrafish as a unique model system in bone research: the power of genetics and in vivo imaging*, Journal of Applied Ichthyology, 26 (2010), pp. 219–224.

[176] N. SRIVASTAVA, G. HINTON, A. KRIZHEVSKY, I. SUTSKEVER, AND R. SALAKHUTDINOV, *Dropout: a simple way to prevent neural networks from overfitting*, The journal of machine learning research, 15 (2014), pp. 1929–1958.

[177] O. STERN, R. MARÉE, J. ACETO, N. JEANRAY, M. MULLER, L. WEHENKEL, AND P. GEURTS, *Automatic localization of interest points in zebrafish images with tree-based methods*, in IAPR International Conference on Pattern Recognition in Bioinformatics, Springer, 2011, pp. 179–190.

[178] V. ŠTIH, L. PETRUCCO, A. M. KIST, AND R. PORTUGUES, *Stytra: an open-source, integrated system for stimulation, tracking and closed-loop behavioral experiments*, PLoS computational biology, 15 (2019), p. e1006699.

[179] D. R. STIRLING, O. SULEYMAN, E. GIL, P. M. ELKS, V. TORRACA, M. NOURSADEGHI, AND G. S. TOMLINSON, *Analysis tools to quantify dissemination of pathology in zebrafish larvae*, Scientific reports, 10 (2020), p. 3149.

[180] U. STRÄHLE, S. SCHOLZ, R. GEISLER, P. GREINER, H. HOLLERT, S. RASTEGAR, A. SCHUMACHER, I. SELDERSLAGHS, C. WEISS, H. WITTERS, ET AL., *Zebrafish embryos as an alternative to animal experiments—a commentary on the definition of the onset of protected life stages in animal welfare regulations*, Reproductive Toxicology, 33 (2012), pp. 128–132.

[181] C. H. SUDRE, W. LI, T. VERCAUTEREN, S. OURSELIN, AND M. JORGE CARDOSO, *Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations*, in Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: Third International Workshop, DLMIA 2017, and 7th International Workshop, ML-CDS 2017, Held in Conjunction with MICCAI 2017, Québec City, QC, Canada, September 14, Proceedings 3, Springer, 2017, pp. 240–248.

[182] S. SUGANYADEVI, V. SEETHALAKSHMI, AND K. BALASAMY, *A review on deep learning in medical image analysis*, International Journal of Multimedia Information Retrieval, 11 (2022), pp. 19–38.

[183] K. SUN, B. XIAO, D. LIU, AND J. WANG, *Deep high-resolution representation learning for human pose estimation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 5693–5703.

[184] M. E. SURYANTO, F. SAPUTRA, K. A. KURNIA, R. D. VASQUEZ, M. J. M. ROLDAN, K. H.-C. CHEN, J.-C. HUANG, AND C.-D. HSIAO, *Using deeplabcut as a real-time and markerless tool for cardiac physiology assessment in zebrafish*, Biology, 11 (2022), p. 1243.

[185] I. SUTSKEVER, J. MARTENS, G. DAHL, AND G. HINTON, *On the importance of initialization and momentum in deep learning*, in International conference on machine learning, PMLR, 2013, pp. 1139–1147.

[186] J. A. SWETS, *Signal detection theory and ROC analysis in psychology and diagnostics: Collected papers*, Psychology Press, 2014.

[187] E. E. TAMM, *Tracking ai's explosive growth in aquaculture*. https://this.fish/blog/ai-guide-tracking-ais-explosive-growth-in-aquaculture/.
Accessed in January, 2025.

[188] M. TARASCO, F. P. CORDELIERES, M. L. CANCELA, AND V. LAIZÉ, *Zfbone: An imagej toolset for semi-automatic analysis of zebrafish bone structures*, Bone, 138 (2020), p. 115480.

[189] M. TARASCO, V. LAIZÉ, J. CARDEIRA, M. L. CANCELA, AND P. J. GAVAIA, *The zebrafish operculum: A powerful system to assess osteogenic bioactivities of molecules with pharmacological and toxicological relevance*, Comparative Biochemistry and Physiology Part C: Toxicology & Pharmacology, 197 (2017), pp. 45–52.

[190] E. TEIXIDÓ, T. R. KIESSLING, E. KRUPP, C. QUEVEDO, A. MURIANA, AND S. SCHOLZ, *Automated morphological feature assessment for zebrafish embryo developmental toxicity screens*, Toxicological Sciences, 167 (2019), pp. 438–449.

[191] S. THAPA AND D. L. STACHURA, *Deep learning approach for quantification of fluorescently labeled blood cells in danio rerio (zebrafish)*, Bioinformatics and Biology Insights, 15 (2021), p. 11779322211037770.

[192] L. S. THOMAS AND J. GEHRIG, *Multi-template matching: a versatile tool for object-localization in microscopy images*, BMC bioinformatics, 21 (2020), pp. 1–8.

[193] T. TIELEMAN, *Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural networks for machine learning, 4 (2012), p. 26.

[194] N. TONACHELLA, A. MARTINI, M. MARTINOLI, D. PULCINI, A. ROMANO, AND F. CAPOCCIONI, *An affordable and easy-to-use tool for automatic fish length and weight estimation in mariculture*, Scientific Reports, 12 (2022), p. 15642.

[195] F. TONELLI, J. W. BEK, R. BESIO, A. DE CLERCQ, L. LEONI, P. SALMON, P. J. COUCKE, A. WILLAERT, AND A. FORLINO, *Zebrafish: a resourceful vertebrate model to investigate skeletal disorders*, Frontiers in endocrinology, 11 (2020), p. 555577.

[196] N. TOROSDAGLI, D. K. LIBERTON, P. VERMA, M. SINCAN, J. S. LEE, AND U. BAGCI, *Deep geodesic learning for segmentation and anatomical landmarking*, IEEE transactions on medical imaging, 38 (2018), pp. 919–931.

[197] N. G. TRILLOS AND R. MURRAY, *A new analytical approach to consistency and overfitting in regularized empirical risk minimization*, European Journal of Applied Mathematics, 28 (2017), pp. 886–921.

[198] N. P. TUCKEY, D. T. ASHTON, J. LI, H. T. LIN, S. P. WALKER, J. E. SYMONDS, AND M. WELLENREUTHER, *Automated image analysis as a tool to measure individualised growth and population structure in chinook salmon (oncorhynchus tshawytscha)*, Aquaculture, Fish and Fisheries, 2 (2022), pp. 402–413.

[199] G. TYAGI, N. PATEL, AND I. SETHI, *A fine-tuned convolution neural network based approach for phenotype classification of zebrafish embryo*, Procedia Computer Science, 126 (2018), pp. 1138–1144.

[200] M. UMAIR, F. AHAMD, M. BILAL, A. ASIRI, M. YOUNUS, AND A. KHAN, *A comprehensive review of genetic skeletal disorders reported from pakistan: a brief commentary*, Meta Gene, 20 (2019), p. 100559.

[201] S. UNGER, C. R. FERREIRA, G. R. MORTIER, H. ALI, D. R. BERTOLA, A. CALDER, D. H. COHN, V. CORMIER-DAIRE, K. M. GIRISHA, C. HALL, ET AL., *Nosology of genetic skeletal disorders: 2023 revision*, American Journal of Medical Genetics Part A, 191 (2023), pp. 1164–1209.

[202] R. VANDAELE, J. ACETO, M. MULLER, F. PERONNET, V. DEBAT, C.-W. WANG, C.-T. HUANG, S. JODOGNE, P. MARTINIVE, P. GEURTS, ET AL., *Landmark detection in 2d bioimages for geometric morphometrics: a multi-resolution tree-based approach*, Scientific reports, 8 (2018), pp. 1–13.

[203] V. VAPNIK, *Principles of risk minimization for learning theory*, Advances in neural information processing systems, 4 (1991).

[204] G. K. VARGHESE, *Understanding the Bias-Variance Tradeoff in Machine Learning*. https://medium.com/@gladinv/understanding-the-bias-variance-tradeoff-in-machine-learning-cf60db555156, 2023.
[Online; accessed 17-July-2024].

[205] Y. VERHAEGEN, D. ADRIAENS, T. DE WOLF, P. DHERT, AND P. SORGELOOS, *Deformities in larval gilthead sea bream (sparus aurata): A qualitative and quantitative analysis using geometric morphometrics*, Aquaculture, 268 (2007), pp. 156–168.

[206] U. VON LUXBURG AND B. SCHÖLKOPF, *Statistical learning theory: Models, concepts, and results*, in Handbook of the History of Logic, vol. 10, Elsevier, 2011, pp. 651–706.

[207] K. VORONTSOV, *Exact combinatorial bounds on the probability of overfitting for empirical risk minimization*, Pattern Recognition and Image Analysis, 20 (2010), pp. 269–285.

[208] A. VOULODIMOS, N. DOULAMIS, A. DOULAMIS, AND E. PROTOPAPADAKIS, *Deep learning for computer vision: A brief review*, Computational intelligence and neuroscience, 2018 (2018), p. 7068349.

[209] A. WALEED, H. MEDHAT, M. ESMAIL, K. OSAMA, R. SAMY, AND T. M. GHANIM, *Automatic recognition of fish diseases in fish farms*, in 2019 14th International Conference on Computer Engineering and Systems (ICCES), IEEE, 2019, pp. 201–206.

[210] C.-W. WANG, C.-T. HUANG, J.-H. LEE, C.-H. LI, S.-W. CHANG, M.-J. SIAO, T.-M. LAI, B. IBRAGIMOV, T. VRTOVEC, O. RONNEBERGER, ET AL., *A benchmark for comparison of dental radiography analysis algorithms*, Medical image analysis, 31 (2016), pp. 63–76.

[211] J. WANG, K. SUN, T. CHENG, B. JIANG, C. DENG, Y. ZHAO, D. LIU, Y. MU, M. TAN, X. WANG, ET AL., *Deep high-resolution representation learning for visual recognition*, IEEE transactions on pattern analysis and machine intelligence, 43 (2020), pp. 3349–3364.

[212] J. WANG, K. WANG, Y. YU, Y. LU, W. XIAO, Z. SUN, F. LIU, Z. ZOU, Y. GAO, L. YANG, ET AL., *Self-improving generative foundation model for synthetic medical image generation and clinical applications*, Nature Medicine, (2024), pp. 1–9.

[213] X. WANG, E. CHENG, I. S. BURNETT, Y. HUANG, AND D. WLODKOWIC, *Automatic multiple zebrafish larvae tracking in unconstrained microscopic video conditions*, Scientific reports, 7 (2017), pp. 1–8.

[214] X. WANG, E. CHENG, I. S. BURNETT, R. WILKINSON, AND M. LECH, *Automatic tracking of multiple zebrafish larvae with resilience against segmentation errors*, in 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), IEEE, 2018, pp. 1157–1160.

[215] V. WEINHARDT, R. SHKARIN, T. WERNET, J. WITTBRODT, T. BAUMBACH, AND F. LOOSLI, *Quantitative morphometric analysis of adult teleost fish by x-ray computed tomography*, Scientific reports, 8 (2018), pp. 1–12.

[216] P. J. WERBOS, *Backpropagation through time: what it does and how to do it*, Proceedings of the IEEE, 78 (1990), pp. 1550–1560.

[217] N. WOJKE, A. BEWLEY, AND D. PAULUS, *Simple online and realtime tracking with a deep association metric*, in 2017 IEEE International Conference on Image Processing (ICIP), IEEE, 2017, pp. 3645–3649.

[218] Z. XU, B. LI, Y. YUAN, AND M. GENG, *Anchorface: An anchor-based facial landmark detector across large poses*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 3092–3100.

[219] D. YAMAMOTO, D. SATO, H. NAKAYAMA, Y. NAKAGAWA, AND Y. SHIMADA, *Zf-mapper: Simple and complete freeware for fluorescence quantification in zebrafish images*, Zebrafish, 16 (2019), pp. 233–239.

[220] 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, 28 (2021), pp. 2785–2816.

[221] H. YAO, Q. DUAN, D. LI, AND J. WANG, *An improved k-means clustering algorithm for fish image segmentation*, Mathematical and Computer Modelling, 58 (2013), pp. 790–798.

[222] B. YEGNANARAYANA, *Artificial neural networks*, PHI Learning Pvt. Ltd., 2009.

[223] Y.-C. YEH, C.-H. WENG, Y.-J. HUANG, C.-J. FU, T.-T. TSAI, AND C.-Y. YEH, *Deep learning approach for automatic landmark detection and alignment analysis in whole-spine lateral radiographs*, Scientific reports, 11 (2021), pp. 1–15.

[224] X. YING, N. J. BARLOW, AND M. H. FEUSTON, *Micro–computed tomography and volumetric imaging in developmental toxicology*, in Reproductive and Developmental Toxicology, Elsevier, 2017, pp. 1183–1205.

[225] X. YING AND T. M. MONTICELLO, *Modern imaging technologies in toxicologic pathology: An overview*, Toxicologic pathology, 34 (2006), pp. 815–826.

[226] J. YOSINSKI, J. CLUNE, Y. BENGIO, AND H. LIPSON, *How transferable are features in deep neural networks?*, in Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, Cambridge, MA, USA, 2014, MIT Press, p. 3320–3328.

[227] J. ZENG, M. FENG, Y. DENG, P. JIANG, Y. BAI, J. WANG, A. QU, W. LIU, Z. JIANG, Q. HE, ET AL., *Deep learning to obtain high-throughput morphological phenotypes and its genetic correlation with swimming performance in juvenile large yellow croaker*, Aquaculture, 578 (2024), p. 740051.

[228] B. ZHANG, K. E. PAS, T. IJASEUN, H. CAO, P. FEI, AND J. LEE, *Automatic segmentation and cardiac mechanics analysis of evolving zebrafish using deep learning*, Frontiers in cardiovascular medicine, 8 (2021), p. 675291.

[229] L. ZHANG, W. LI, C. LIU, X. ZHOU, AND Q. DUAN, *Automatic fish counting method using image density grading and local regression*, Computers and Electronics in Agriculture, 179 (2020), p. 105844.

[230] X. ZHIPING AND X. E. CHENG, *Zebrafish tracking using convolutional neural networks*, Scientific reports, 7 (2017), p. 42815.

[231] C. ZHOU, K. LIN, D. XU, J. LIU, S. ZHANG, C. SUN, AND X. YANG, *Method for segmentation of overlapping fish images in aquaculture*, International Journal of Agricultural and Biological Engineering, 12 (2019), pp. 135–142.

[232] Z.-H. ZHOU, *Ensemble methods: foundations and algorithms*, CRC press, 2012.

[233] X. ZHU AND A. B. GOLDBERG, *Introduction to semi-supervised learning*, Springer Nature, 2022.

[234] X. J. ZHU, *Semi-supervised learning literature survey*, (2005).

[235] F. ZHUANG, Z. QI, K. DUAN, D. XI, Y. ZHU, H. ZHU, H. XIONG, AND Q. HE, *A comprehensive survey on transfer learning*, Proceedings of the IEEE, 109 (2020), pp. 43–76.

[236] L. I. ZON AND R. T. PETERSON, *In vivo drug discovery in the zebrafish*, Nature reviews Drug discovery, 4 (2005), pp. 35–44.

[237] İLYUREK KILIÇ, *Understanding AUC - ROC Curve*.

https://medium.com/@ilyurek/roc-curve-and-auc-evaluating-model-performance-c2178008b02, 2023.
[Online; accessed 26-July-2024].