

Fred Intwaza Kwizera

Mutual Information Maximization by Data Augmentation for Plankton Classification

Master's thesis in Cybernetics and Robotics

Supervisor: Annette Stahl

June 2023

Fred Intwaza Kwizera

Mutual Information Maximization by Data Augmentation for Plankton Classification

Master's thesis in Cybernetics and Robotics
Supervisor: Annette Stahl
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics



Abstract

This thesis explores the potential of unsupervised machine learning methods for in-situ plankton image classification. The research was driven by the need to overcome the limitations of supervised methods, which necessitate extensive annotated datasets and struggle with classifying unseen plankton. Employing Invariant Information Clustering (IIC) and two variants of Regularized Information Maximization (RIM), outperforming previous unsupervised techniques by 11%, reaching an accuracy of 24.9% on the NDSB dataset. A novel variant of RIM using selective image transformations for augmentation was introduced, surpassing the performance of the IMSAT method.

Normalized Mutual Information (NMI) proved to be an effective evaluation metric for unsupervised methods when dataset labels are not available. Despite computational constraints and the assumption of known dataset classes, the findings imply a substantial potential for unsupervised machine learning methods in plankton image classification.

Application for in-situ classification necessitates an additional manual post classification step by a plankton expert to label clusters. Despite these limitations, the results suggest promising avenues for future research. The presented methods mark an advancement in the field of plankton image classification, with potential benefits for marine ecosystem monitoring and the study of environmental impacts.

Sammendrag

Denne masteroppgaven utforsker potensialet for ikke-veiledet maskinl ring for in-situ plankton bildeklassifisering. Oppgaven ble utformet for   adressere begrensningene i bildeklassifisering ved veiledet maskinl ring, hovedsakelig behovet for et omfattende annotert datasett og utfordringen med   klassifisere plankton som ikke er til stede i treningssettet. Ved   benytte Invariant Information Clustering (IIC) og to varianter av Regularized Information Maximization (RIM), overgikk studiet tidligere ul rte teknikker med 11%, og oppn dde en nøyaktighet p  24.9% p  NDSB-datasettet. En ny variant av RIM som bruker selektive bildetransformasjoner for augmentering ble introdusert, og overgikk ytelsen til IMSAT metoden.

Normalized Mutual Information (NMI) viste seg   v re en effektiv evalueringsmetrikk for ikke-veiledet metoder n r datasettmerkene ikke er tilgjengelige. Til tross for beregningsbegrensninger og antagelsen om kjente datasettklasser, indikerer funnene betydelig potensiale for ul rte maskinl ringsmetoder i plankton bildeklassifisering.

Applikasjon for in-situ klassifisering krever et ekstra manuelt trinn etter klassifisering av en planktonekspert for   merke de klassifiserte gruppene. Til tross for disse begrensningene, antyder resultatene lovende veier for fremtidig forskning. De presenterte metodene markerer fremgang innen feltet for plankton bildeklassifisering, med potensielle fordeler for overv king av marine  kosystemer og studier av milj p virkninger.

Preface

This thesis is submitted as a part of the requirements for the master's degree at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology. The work presented in this thesis has been carried out under the supervision of Prof. Annette Stahl from the same department.

The work conducted in the thesis is a continuation of AILARON: Autonomous Imaging and Learning Ai ROBOT identifying plaNkton taxa in-situ. This spring, my engagement with the AILARON project has involved familiarizing myself with the project objectives and challenges, doing a literature review, selecting appropriate unsupervised machine learning approaches, exploring and preprocessing datasets, and implementing the proposed methods.

The computational resources for this work were provided by the Faculty of Information Technology and Electrical Engineering, to whom I express my gratitude. Their contribution was instrumental in achieving the computational heavy tasks associated with the applied machine learning and image classification methods. Lastly, I wish to express my heartfelt gratitude to my mother for her unwavering inspiration and support.

I hope this thesis provides valuable insights into the potential of unsupervised machine learning methods for in-situ plankton image classification.

Contents

Abstract	ii
Sammendrag	iii
Preface	iv
Contents	v
Figures	vii
Tables	viii
1 Introduction	1
1.1 Background and Motivation	1
1.1.1 Research Questions and Objectives	2
1.2 Contribution	3
1.3 Scope and Limitations	3
1.4 Outline of the Thesis	3
2 Theory	5
2.1 Plankton	5
2.2 Fundamentals of Statistics	7
2.3 Mutual Information	9
2.4 Machine Learning Fundamentals	10
2.4.1 Unsupervised Machine Learning	12
2.5 Image Augmentation	14
3 Literature Review	18
3.1 Plankton Sampling	18
3.2 Existing Plankton Datasets	20
3.3 Plankton Classification	23
3.4 Mutual Information Maximization	24
4 Methodology	26
4.1 Dataset Selection and Preprocessing	26
4.2 Loss Function	27
4.2.1 Machine Learning Models	29
4.3 Evaluation	30
5 Implementation	32
5.1 Hardware and Software Setup	32
5.2 Dataset Preprocessing	32
5.3 Loss Function	33
5.4 Machine Learning Models	35

5.5	Training and Evaluation	36
6	Results and Discussions	39
6.1	Dataset Exploration and Preprocessing	39
6.1.1	Unsupervised Classification	41
6.1.2	Image Augmentations	44
6.2	Limitations of the Study	44
6.3	Applicability for AILARON	47
7	Conclusion	48
8	Future Work	49
	Bibliography	50
A	Mathematical Expressions and Derivations	55
A.1	Approximating the Marginal Distribution	55
A.2	Entropy	55
A.3	Mutual Information	56
A.4	Loss Functions	57
A.4.1	Information Maximizing Self-Augmented Training (IMSAT)	57
B	Dataset Distribution	60
C	Hyperparameters	63

Figures

2.1	Examples of Zooplankton	6
2.2	Size groups of plankton	7
2.3	A harmful algae bloom	8
2.4	Venn diagram illustrating the concept of mutual information	10
2.5	Categories of machine learning	11
2.6	Machine Learning Architectures	13
2.7	Example of image transformation, including color transformation and Gaussian blur	16
3.1	Underwater devices for plankton image sampling	19
3.2	Example images from the NDSB dataset	21
3.3	Example images from the AILARON dataset	22
4.1	An illustration of Regularized information Maximization (RIM)	27
4.2	An illustration of Invariant Information Clustering (IIC)	29
5.1	Diagram of the machine learning and evaluation process	37
6.1	Density plot of the dataset distribution, NDSB dataset	40
6.2	Variations in image format, quality, and class variability	41
6.3	Confusion Matrix of Classifier trained with IIC.	44
6.4	Classifier training using IIC and IMSAT	45
6.5	Examples of the image augmentations	46
B.1	The distribution of the AILARON dataset. Some classes do not con- tain samples. The dataset contains non-taxonomic classes.	61
B.2	The distribution of the DNSB dataset. The distribution is skewed, showing a large number of instances for few classes.	62

Tables

2.1	Examples of Affine transformations, including scaling, rotation, and reflection	15
5.1	Python libraries used in the study	33
5.2	Parameters of the classifier model architectures	36
6.1	Distribution of ten selective classes.	42
6.2	Classifier performance (%) for the NDSB dataset.	42
6.3	Classifier performance for a subset of ten classes for the NDSB dataset	43
C.1	Batch sizes for machine learning architectures.	63
C.2	Hyperparameters applied during training. Commonly reported values were used to minimize application-specific tuning.	63

Chapter 1

Introduction

Plankton is of great importance to marine ecosystems and the climate in general. As the primary food source for many higher trophic-level organisms, they also produce a considerable portion of the world's oxygen. Given their sensitivity to environmental changes, monitoring plankton is crucial to assess the impacts of climate change, pollution, and other stressors. Recent computer vision and robotics advancements have enabled continuous spatial and temporal plankton observations using autonomous underwater vehicles. The current work aims to demonstrate how unsupervised machine learning can be applied for in-situ plankton image classification.

1.1 Background and Motivation

Conventional methods for monitoring the abundance and diversity of plankton require considerable resources. For instance, the monitoring program Continuous Plankton Recorder (CPR) is world's the most expensive long-term survey of marine organisms. Similar ship-based water-sampling approaches are done at discrete locations and therefore lack high spatial and temporal resolution. Furthermore, subsequent classification imposes time-consuming manual effort on taxonomists (Reid et al., 2003; Saad et al., 2020). The lack of spatial and temporal resolution is further exacerbated by the seasonal variations in the plankton taxa (Haug et al., 2021b), plankton's high sensitivity to small variations in the environment (Salvesen et al., 2020), and a large number of planktonic species and inter-species variations and mutations.

Recent advancements in computer vision and robotics have shifted water-based sampling and manual labeling toward image-based sampling and automated classification, for instance in Cowen and Guigand, 2008; Davies et al., 2017; Olson and Sosik, 2007; Saad et al., 2020. These approaches rely on ships or use Autonomous Underwater Vehicles (AUV) for sampling. Subsequent classification often relies on supervised machine learning (ML) approaches, necessitating large annotated datasets. To circumvent the need for annotated datasets, a group of unsupervised machine-learning approaches based on representation learning and clus-

tering have been proposed in Salvesen et al., 2020, 2022. Despite these efforts, an effective unsupervised method for plankton image classification is yet to be achieved. This study suggests a second group of unsupervised machine learning methods based on mutual information maximization and image augmentation for grouping images of similar plankton.

1.1.1 Research Questions and Objectives

This thesis is a continuation of AILARON¹ (Saad et al., 2020), a mobile robotic tool for studying microbial life in the upper water column. Combining imaging, processing, analysis, and classification of plankton-based imagery, AILARON enables targeted sampling. The classifier of AILARON is based on supervised machine learning, making it unable to assign meaningful labels to plankton that are not present in the training set, i.e., it cannot "discover" unseen plankton. The research questions of the thesis are formulated to address this issue.

- Can the proposed unsupervised machine learning methods effectively group images of similar plankton?
- How well does the chosen unsupervised evaluation metric coincide with the accuracy of the classifier?
- What image augmentation techniques are effective in improving the performance of the classifier?
- How does the performance of the proposed methods compare to other unsupervised machine learning methods?
- Are the proposed methods applicable to AILARON?

The research questions will be addressed by considering end-to-end (an architecture where no feature selection and extraction prior to clustering is necessary) unsupervised ML methods for training a plankton image classifier. Three methods for learning a classifier will be considered: Invariant Information Clustering (IIC), a method that aims to learn a classifier by maximizing the mutual information between an image and its augmentation, and two variations of Regularized Information Maximization (RIM), a method that aims to learn a classifier by maximizing mutual information between images and their labels through self-augmentation. The two variations employ different augmentation techniques: Virtual Adversarial Training (VAT) applies a local perturbation to an image to obtain its augmentation and a novel approach where selective image transformations are applied. Considering the second research question, when training and evaluating a classifier in an unsupervised manner, the labels of the dataset are unavailable. To assess the performance of a classifier, alternative metrics to accuracy must be considered. This thesis will assess Normalized Mutual Information (NMI) as an unsupervised performance metric. The proposed methods will be evaluated in two cases, one considering a large plankton dataset of 121 classes, and a second

¹ AILARON stands for Autonomous Imaging and Learning Ai RObot identifying plaNkton taxa in-situ

case considering a subset of ten distinct classes. K-fold cross-validation will be used to assess the classifier performance for the two cases. The objectives of the thesis are formulated according to these methods and the research questions.

1. Implement and evaluate (using ground truth labels) the performance of the proposed methods.
2. Compare Normalized Mutual Information to Accuracy as a performance metric.
3. Identify image augmentation techniques that effectively improve the classifier's performance.
4. Benchmark the proposed method against existing methods.
5. Considering the hardware limitations, investigate whether the methods are applicable to AILARON.

1.2 Contribution

This thesis presents a classifier using IIC and the Inception architecture, achieving a 10% performance improvement over previously reported unsupervised approaches on the same dataset. Additionally, the thesis introduces a novel variant of RIM that uses selective image transformation for augmentation.

1.3 Scope and Limitations

This thesis will not address the stages preceding image classification, specifically object detection, and segmentation. In addition, when training and evaluating the proposed methods, the number of classes within the dataset is assumed to be known. Furthermore, it is important to note that the proposed methods are trained on a specific dataset. Therefore, due to variations in plankton taxa, the results might not be universally applicable to all plankton datasets. On a final note, supervised machine learning has been thoroughly researched and implemented across various domains, with its performance generally well-documented. Conversely, while unsupervised machine learning is increasingly recognized as a significant research area, it presents significant challenges.

1.4 Outline of the Thesis

The thesis is structured as follows. Chapter 2 describes the concepts used throughout the thesis: an introduction to plankton, necessary prerequisites in statistics, mutual information, and image augmentation. Chapter 3 presents previous studies on plankton sampling, classification, and mutual information maximization. In addition, an overview of some existing plankton datasets is provided. Chapter 4 details the methodology of the thesis, including the loss function of the proposed methods and the selection of classifier models. Chapter 5 covers the hardware

and software setup, the implementation of the loss functions, and the training and testing of the classifier. Chapter 6 presents and discusses the results of the thesis, while Chapter 7 concludes the thesis by summarizing the key findings.

Chapter 2

Theory

This chapter provides the theoretical basis for understanding the concepts and methods applied in this thesis. The chapter is divided into four parts. First, we will explain the important role of plankton in its ecosystem and the climate in general in Section 2.1. Thereafter, key statistical concepts relevant to the study are explained in Section 2.2 and Section 2.3. Finally, the prerequisites of machine learning are considered in Section 2.4 before providing an overview of some image augmentation techniques in Section 2.5.

2.1 Plankton

Plankton refers to a diverse range of microorganisms found in water bodies incapable of moving against currents or winds (Carol Lalli and Timothy Parsons, 1997). These organisms can vary significantly in size, feeding habits, ecological functions, life cycle, environmental sensitivities, and other characteristics (Chust et al., 2017). Depending on whether the organism is an animal or a plant, plankton is broadly categorized into two groups, *phytoplankton* and *zooplankton*. Though plankton is often associated with smaller organisms, zooplankton include larger organisms such as jellyfish of several meters in diameter (Carol Lalli and Timothy Parsons, 1997). Notably, not all plankton are passive; some plankton, such as copepods, can swim several meters over a day (Svetlichny et al., 2020). Though plankton can be grouped by a wide range of characteristics, we will consider these organisms in terms of size. Some size groups of plankton are *megaplankton*, *macroplankton*, *mesoplankton* and *microplankton* (Makoto and Tsutomu, 1984)¹.

Plankton performs several crucial functions. Firstly, they are a vital part of the base marine food chain. Zooplankton, which feeds on other zooplankton and phytoplankton, serves as a food source for small and large organisms such as krill, small fish, jellyfish, whales, and birds (Lampert and Sommer, 2007). Second, phytoplankton performs photosynthesis, absorbing carbon dioxide from the atmosphere and producing oxygen. It is estimated that phytoplankton is responsible for 50% of

¹ Some size groups are irrelevant for this thesis and are therefore excluded.



Figure 2.1: Examples showing common Zooplankton taxa, including Copepods, Calanidae Emora Stylifera, and more. From Zingone et al., 2019 sampled in the Gulf of Naples, Italy. Source: [Zooplankton n.d.](#)

the world's oxygen production. Complementary, plankton is important in the biological pump by transporting large quantities of carbon dioxide from the surface to the deep ocean and sediments (Rost et al., 2008). Lastly, planktonic organisms are of great commercial importance for the world's fisheries and aquaculture industries, for which the livelihood of many people depends. For instance, it serves as the primary diet for several commercially valuable fish species, especially during their larval stage. The abundance of plankton directly impacts these fish species' distribution, growth, and thus overall fishery productivity and commercial yield (TR Parsons and CM Lalli, 2002. Paerl, Gardner et al., 2016; Paerl and Huisman, 2008).

Conversely, plankton can, in some cases, pose a hazard to the environment and humans. Certain plankton species can accumulate in large numbers, causing what is often referred to as *algae blooms* Hallegraeff et al., 2004. A subset of these species can produce potent toxins that can accumulate in fish and shellfish, posing a risk to human consumers. Various research supports that the abundance and diversity of plankton are sensitive to environmental changes (Barton et al., 2013; Rost et al., 2008), and algae blooms are often caused by temperature or other environmental fluctuations (Paerl and Huisman, 2009). Some research further suggests that these blooms are exacerbated by climate change (Paerl, Gardner et al., 2016;

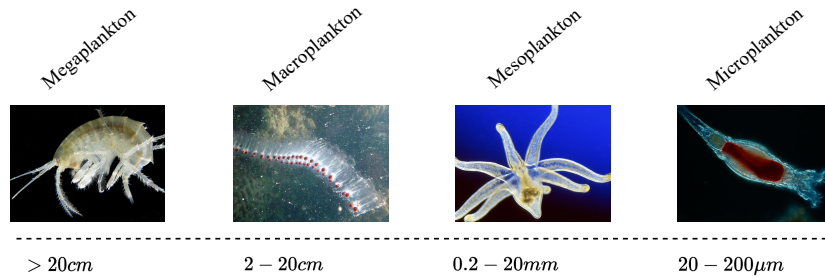


Figure 2.2: The size groups of plankton spanning from more than 20cm to 20µ, including Amphipoda, Thaliacea, Tube Anemone larva, and Syndermata from left to right. Source: *Gammarus roeselii* n.d.; *Meroplankton* n.d.; *Mikrofoto.de-Raedertier-14* n.d.; *Salps Thaliacea* n.d.

Paerl and Huisman, 2008).

2.2 Fundamentals of Statistics

This section explains concepts in probability and statistics applied throughout this thesis. It serves as supporting literature for comprehending the unsupervised learning methods to be introduced in the subsequent chapters. It is assumed that the reader has a basic understanding of the following concepts: discrete and continuous random variables, joint probability, conditional and marginal probability, expected value, probability mass function and probability density function, basic probability distributions, and Bayesian inference. No attempt will be made to explain these concepts, so readers are advised to consult external sources if needed. The concepts covered in this section include entropy, the Kullback-Leibler divergence, and mutual information.

Entropy

Entropy² is a measure of uncertainty or randomness related to a random variable or probability distribution (Shannon, 2001). It quantifies the average amount of information or "surprise" contained in the possible outcomes of the variable. In other words, entropy represents the average amount of information required to describe the outcome of a random event. For a discrete random variable X with distribution p , the entropy is defined as (Shannon, 2001)

$$H(X) := - \sum_{x \in \mathcal{X}} p(x) \log p(x). \quad (2.1)$$

²The term entropy is used for different concepts in various fields. These concepts are commonly associated with uncertainty. In this context, we solely referred to entropy in information theory.



Figure 2.3: A harmful algae bloom in Lake Erie, North America, in September 2017. The bloom covered over 1800 square kilometers. Source: *Lethal algae blooms an ecosystem out of balance* n.d.

The base of the logarithm function is chosen based on the specific application. To illustrate the concept of entropy, consider the scenario of guessing a coin flip. Suppose there are two coins: a fair coin and an unfair coin (with a higher probability of landing on, e.g., tails). When flipping the fair coin, there is more uncertainty of the outcome because both heads and tails are equally likely, resulting in higher entropy. In contrast, flipping the unfair coin allows for a better guess if the bias is known, indicating lower uncertainty or entropy associated with the unfair coin.

Kullback–Leibler Divergence

The Kullback-Leibler (KL) divergence, also known as relative entropy, quantifies the dissimilarity between two probability distributions (Kullback and Leibler, 1951). It measures how one probability distribution differs from another reference distribution. The KL divergence is always non-negative and equals zero only when the two distributions are identical. For two discrete probability distributions P and Q of a random variable X , the KL divergence from P to Q is defined as:

$$D_{\text{KL}}(P|Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right). \quad (2.2)$$

Consider the illustrative scenario where we have an English book and its translated version in French. In this case, each book can be seen as a 'source' that generates a sequence of characters, and we can treat these sequences as outcomes of a random variable. By counting the occurrences of each character in both books, we can

derive two probability distributions. These distributions represent the likelihood of individual characters appearing in the English and French texts, respectively. In essence, the KL divergence provides a numerical value that reflects the dissimilarity between the character frequency distributions of the two books. Note that the KL divergence is not symmetrical (i.e., the divergence from the English distribution to the French one is not the same as the divergence from the French to the English).

2.3 Mutual Information

Mutual information (MI) is a special case of KL divergence and expresses the dependency between two random variables. It quantifies the amount of information you can gain about one variable, given the knowledge of another (Cover, 1999). This is an assessment of how the uncertainty or unpredictability about one variable decreases when the other variable is known. Formally, given two random variables X and Y with joint probability distribution $P_{X,Y}$ and marginal probability distributions P_X and P_Y respectively, mutual information is defined as:

$$I(X; Y) = D_{\text{KL}}(P_{(X,Y)} | P_X \otimes P_Y). \quad (2.3)$$

Here, \otimes represents the tensor product of the resulting variables (Cover, 1999). To illustrate the concept, consider the relationship between a person's choice of outdoor clothing and the weather. We can observe that a person prefers wearing shorts on sunny days, pants on cloudy days, and a raincoat when rainy. Therefore the choice of clothing depends on the weather condition. In this scenario, knowledge about the weather condition provides valuable information about the person's clothing choice. This relationship or MI between the weather and clothing choice can be quantified using Equation (2.3). In comparison, KL divergence and MI are measures from information theory that quantify some aspect of 'difference' or 'dependence'; they do so in subtly different ways. While KL divergence measures how one probability distribution diverges from a second, expected distribution, MI measures the reduction in uncertainty about one random variable given knowledge of another.

Considering the discrete case, in terms of Probability Mass Function (PMF), the MI between two discrete variables is described in Equation (2.4). The derivation for MI in terms of PMFs is presented in Appendix A.3.

$$I(X; Y) = \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} P_{(X,Y)}(x, y) \log \left(\frac{P_{(X,Y)}(x, y)}{P_X(x)P_Y(y)} \right) \quad (2.4)$$

The joint distribution of variables X and Y is denoted by $P_{(X,Y)}$, while $P_X(x)$ and $P_Y(y)$ represent the marginal distributions of X and Y , respectively. An alternative formulation of MI can also be obtained by employing the concept of entropy. This can be illustrated using a Venn diagram as in Figure 2.4. The derivation for MI in terms of entropy is presented in Appendix A.2. Let $H(Y)$ denote the entropy of

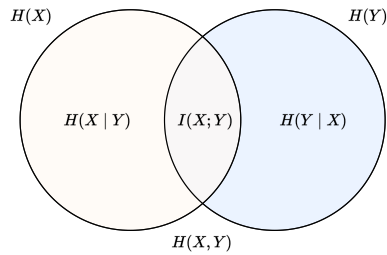


Figure 2.4: A Venn diagram illustrating the concept of mutual information. Each circle represents the set of all possible outcomes for a specific variable expressed in entropy. The overlapping area represents the mutual information between the variables, indicating the shared knowledge gained about one variable through knowing the other.

the stochastic variable Y , and $H(Y | X)$ denote the conditional entropy, MI is then expressed as:

$$I(X; Y) = H(Y) - H(Y | X). \quad (2.5)$$

2.4 Machine Learning Fundamentals

Machine learning (ML) is a field of Artificial Intelligence (AI) that uses statistical concepts to enable computers (hereafter referred to as the ML model) to learn from data without being explicitly programmed. The ML applications are many, spanning industries and disciplines from healthcare to finance, autonomous vehicles, and natural language processing, among others. This section will present the following: a brief explanation of the categories of machine learning, an overview of the building blocks of an ML model, an explanation of how ML models learn and are evaluated, and central ML building blocks for computer vision applications. Emphasis will be placed on the field of unsupervised ML. Note that the explanation of basic ML concepts will be brief. The reader is encouraged to seek external material on these topics for further elaboration. The explanations of the concepts described in this section are cited from Goodfellow et al., 2016

Different Machine Learning Approaches

ML approaches can broadly be divided into three broad categories: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning, as illustrated in Figure 2.5. Supervised learning involves training an ML model on labeled data. Types of supervised machine learning include active learning (where the model actively selects specific data points to learn from), regression (where the output space of the model is continuous), and classification (where the model predicts a categorical outcome). Conversely, unsupervised learning aims to find inherent patterns in the unlabeled dataset. Semi-supervised learning is

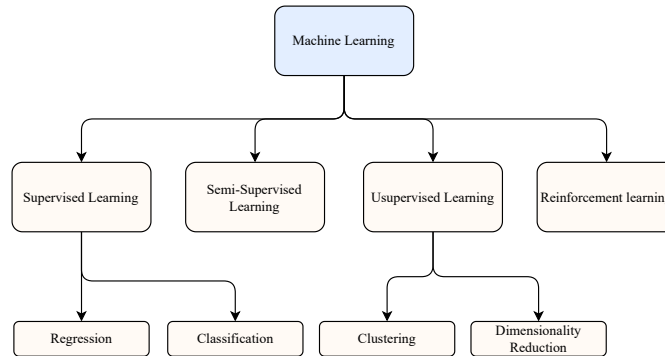


Figure 2.5: The broad categories of machine learning. The two main types of supervised machine learning include regression and classification. Clustering and dimensionality reduction are typical examples of unsupervised learning.

a middle ground that uses a combination of labeled and unlabeled data. Lastly, reinforcement learning allows an agent to learn through interactions with its environment, aiming to maximize some reward function.

A Vanilla Neural Network

The fundamental building blocks of a Machine Learning (ML) model include features, neurons, layers, activation functions, and normalization layers. *Features* denote some properties or characteristics used as input to an ML model. In computer vision, this could be the pixel values of an image. A *neuron* (also called an artificial neuron or node) is a concept inspired by the human brain and describes a function that receives one or more inputs. This function is typically expressed as a non-linear *activation function* applied on a weighted sum of its inputs. The output can then be passed on as input to neurons in the following network layer. The activation function introduces non-linearity into the model, enabling it to learn complex patterns. Common activation functions are the sigmoid activation functions and the ReLU activation function. In simple terms, neurons that take (some) the same inputs form a layer, and layers of neurons form an *artificial neural network* (ANN) model. Several types of layers can be applied to improve model performance. E.g., normalization layers set the distribution of the output of one layer to have a zero mean and unit variance. The neurons' weights and normalization layer constitute the learnable parameters of the model.

Building Blocks for Image Classification

Specific to the application of image classification, certain specialized layers and techniques are commonly used, including convolution layers, pooling layers, and dropout layers. Convolution layers are foundational to Convolutional Neural Networks (CNNs), applying filters to the input data to extract fundamental features

like edges and textures. Following these, pooling layers are often used to decrease the spatial dimensions of the input, thereby reducing the computational load. This dimension reduction is achieved through techniques like Max Pooling or Average Pooling.

Backpropagation

The learning of a model follows an iterative optimization process (a set of *epochs*) by a set of forward and backward passes (*backpropagation*). During the forward pass, the model takes the input data (or a subset where the size is specified by the *batch size*) and makes a prediction based on the current parameters. The predictions and actual target values of the input data are compared according to the loss function, indicating the performance of the current model. The loss *gradient* is calculated w.r.t the model parameters in the backpropagation step. This process is posed as an optimization problem, and common algorithms applied are gradient descent and stochastic gradient descent (depending on whether the data is divided into batches). The gradient represents the direction in which the tuning of the parameters must be made to improve the model predictions (performance according to the loss function). *Overfitting* occurs when the model learns the training data too well and performs poorly on unseen data. *Hyperparameters*, which are parameters not learned from the data, govern the learning process and can significantly impact model performance. These parameters are often obtained by testing different values or using dedicated methods.

Performance Measure

Accuracy is a common metric for evaluating the performance of a classifier. It is defined as the ratio of the number of correct predictions to the total number of predictions made. Accuracy is a suitable metric when the classes are balanced, meaning there are approximately equal numbers of samples for each class. However, in cases where the dataset is imbalanced, with one class having significantly more samples than others, accuracy can be misleading. In such scenarios, a model could achieve high accuracy by merely predicting the majority class, while failing to make meaningful predictions for the minority class. *Balanced accuracy* is an alternative metric that takes into account the class distribution of the dataset. In simple terms, balanced accuracy calculates the accuracy for each class individually and then takes the average.

2.4.1 Unsupervised Machine Learning

Unsupervised machine learning in computer vision analyzes visual data (images or videos) without pre-provided labels or specific supervision. It is advantageous when the manual annotation is impractical and costly or when discovering unseen patterns is desired. The three ML architectures illustrated in Figure 2.6 will be considered in the following.

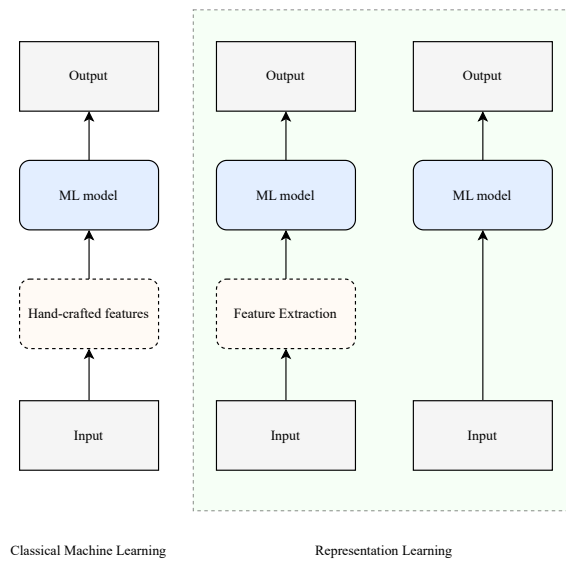


Figure 2.6: Three common architectures in computer vision: handcrafted features, feature learning, and end-to-end. Handcrafted features capture predefined aspects of images, while feature learning employs ANNs. End-to-end learning models directly learn hierarchical features from raw input data.

Classical Machine Learning

Earlier approaches in computer vision often involved handcrafted features. These are feature descriptors explicitly designed and engineered by researchers to capture certain aspects of an image. The features were usually based on insights about the most useful image characteristics for the task, such as corners, edges, and color histograms. Examples of handcrafted features include Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), and Histogram of Oriented Gradients (HOG) (Szeliski, 2022).

Representation Learning

With the rise of deep learning, there was a shift towards learning features directly from data rather than hand-engineering them. One approach in this category is using autoencoders, a type of neural network that learns to encode input data into a set of features and then decode these features to reconstruct the input. Autoencoders can be used for dimensionality reduction, anomaly detection, and denoising tasks. Such methods automatically learn hierarchical representations of the data, capturing complex patterns and structures that might be missed with handcrafted features. More recently, end-to-end learning approaches have become popular. These methods aim to learn a mapping from raw input data to output predictions in a single model without manual feature engineering or selec-

tion. Convolutional Neural Networks (CNNs) are the most well-known example of end-to-end learning in computer vision. CNNs learn to extract hierarchical features from raw pixel data, with lower layers learning to detect simple patterns like edges and colors and higher layers learning to recognize more complex structures like objects or scenes. Other examples of end-to-end architectures include Recurrent Neural Networks (RNNs) for sequence data, Transformer models for tasks requiring attention mechanisms, and Generative Adversarial Networks (GANs) for generating new data samples.

Unsupervised Loss Function

A key challenge in this field is defining what it means for two data points to be "similar". The criteria for this can vary widely between different methods and applications. One common strategy is geometric, including k-means clustering or hierarchical clustering, where the focus is on quantifying distances or densities among data points. A second approach is statistical, examining correlations or dependencies between variables. One example is the maximization of mutual information, which quantifies how much information is gained about one variable through observing another. Lastly, reconstructive methods are often utilized in techniques such as autoencoders, where similarity is assessed based on the ability of learned representations to reconstruct original input data.

2.5 Image Augmentation

In machine learning, *data augmentation* is a strategy applied to datasets to enhance model performance (Perez and Wang, 2017). This technique involves generating new instances by applying transformations to existing ones, thereby increasing the dataset's size and diversity. This can help reduce the potential for overfitting (Perez and Wang, 2017). In unsupervised ML, augmentation techniques are frequently used in a paradigm known as contrastive learning. Contrastive learning involves creating augmented versions of the same image (positive pairs) and different images (negative pairs). The goal is to make the representations of the positive pairs similar and the representations of the negative pairs dissimilar, which aids the model in learning discriminative features. Moreover, image augmentation techniques can be implemented to reinforce the intended variance in the data Dosovitskiy et al., 2014. For instance, in an unsupervised setting, the ML model can become invariant to orientation by applying rotations to dataset instances. The following section will present commonly used image transformations for computer vision applications.

Affine Transformations

A set of techniques commonly used for image augmentation are affine transformations. These augmentations are geometric transformations that preserve lines and

parallelism Szeliski, 2022. In simple terms, when applying a transformation to an image, points that are on a line in the original image will remain on a line in the transformed image. Similarly, parallel lines in the original image will remain parallel in the transformed image. For the pixel coordinates in an image (x, y) , an affine transformation can be expressed as

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The applied transformation is determined by the elements of the A matrix. Common affine transformations include scaling, rotation, and reflection. Examples of these rotations, along with their respective A matrices, are presented in Table 2.1.





Transformation Name	Affine Matrix, A	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Rotation	$\begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Reflection	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	

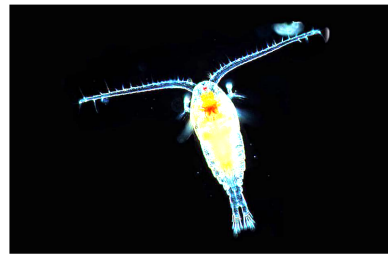
Table 2.1: The table shows examples of the Affine transformations, scaling, rotation, and reflection, along with their respective A matrix. The transformed image can be obtained by applying a vector multiplication between each pixel of the original image with the respective A matrix. In the calculation, the pixel coordinates i take the form $[x_i, y_i, 1]^T$. Source: *Calanoida* n.d.

Color Transformations

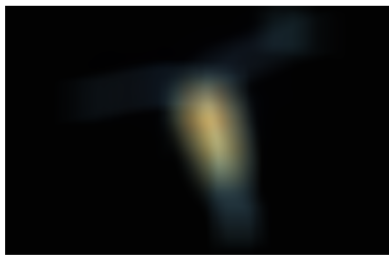
Color image processing can be divided into two main categories: *pseudo-color* and *full-color* processing (Gonzalez, 2009). Pseudo-color processing involves mapping grayscale intensities to colors, essentially adding a color representation to what was originally a black-and-white image. On the other hand, full-color processing deals with images that already have color information, utilizing extended color models such as RGB. While color image processing encompasses a broad range of techniques, in this thesis, we will focus on a basic technique used in pseudo-color processing, specifically the concept of *intensity*. The intensity in a grayscale image refers to the aggregate brightness of the pixel values. Typically, the pixel value in a grayscale image ranges from 0 (indicating black) to a maximum value of 255 (indicating white). Adjusting the brightness of the image can be achieved by adding a specific value to each pixel. The resulting effect on image brightness is illustrated in Figure 2.7b.



(a) Original image.



(b) Adjusted brightness.



(c) Gaussian blurring.



(d) Adjusted sharpness.

Figure 2.7: The figures illustrate various transformations applied to a copepod image (a). (b): Image with brightness factor applied to each color channel. (c): results from a Gaussian convolution operation, and (d) is derived by subtracting a blurred version from the original image. Source: [Calanoida n.d.](#)

Kernel Based Filtering

Kernel-based filters, also known as convolution filters, are a type of linear filter that apply a linear transformation to a pixel and its surrounding pixels to obtain the pixel value in the transformed image (Szeliski, 2022). The transformation is specified by a kernel (Szeliski, 2022). In simple terms, the kernel "slides" over the image, and at each position, it performs a convolution operation between the kernel and a patch of the image. For an RGB image, this is done for each image channel. Consider the pixel at position (x, y) . The pixel value in the transformed image is obtained by:

$$g(x, y) = h * f(x, y) = \sum_{dx=-a}^a \sum_{dy=-b}^b h(dx, dy) f(x - dx, y - dy).$$

$f(x, y)$ and $g(x, y)$ represent the pixel values of the original and transformed image, respectively, and h is the filter kernel. The size of the filter kernel is specified by parameters a and b . One common kernel applied in image processing is the *Gaussian filter*, which smooths or blurs an image. An example is illustrated in Figure 2.7c. The kernel of a Gaussian filter h_g , is specified by the Gaussian function, and the degree of blurring is determined by the size of the kernel, dx , and dy , and the standard deviation of the Gaussian function, σ . A smaller sigma leads to a narrow distribution and less blurring as it gives more weight to the central pixel over its neighbors. Conversely, a larger sigma results in a wider distribution, thus increasing the blur.

$$h_g = g(dx, dy) = \frac{1}{2\pi\sigma^2} e^{-\frac{dx^2+dy^2}{2\sigma^2}}$$

Chapter 3

Literature Review

Section 3.1 and Section 3.3 provides the historical context of plankton sampling and classification and the transformative role of computer vision and computer science in automating these processes. In addition, an overview of some existing plankton datasets is presented in Section 3.2. Lastly, in Section 3.4, the current research on the proposed methods, mutual information maximization for classification, will be reviewed.

3.1 Plankton Sampling

The study and classification of plankton have held significant importance in biological and ecological research, with efforts dating back to the 19th century. Notably, the Continuous Plankton Recorder (CPR) (Reid et al., 2003) marked an important advancement in the study of near-surface plankton taxa. The CPR survey is part of the Global Ocean Observing System (GOOS), a comprehensive global initiative for ocean observation Moltmann et al., 2019. CPR uses a specialized device for plankton sampling, depicted in Figure 3.1c. The device is towed to ships to collect samples from the ocean surface water. These samples are then transported to laboratories for assessments by taxonomists, imposing time-consuming manual labor.

Advancement in computer vision has made possible the transition from water-based sampling methods to image-based sampling. In Olson and Sosik, 2007, a specialized device towed by a vessel captures images of mesoplankton using the ichthyoplankton imaging system (ISIIS). The ISIIS system, depicted in Figure 3.1d, employs a shadowgraph technique in which a light beam is projected through the water, and planktonic organisms in its path cast shadows onto a high-resolution digital camera sensor. In Davies et al., 2017, a particle imaging system is proposed for sampling particles in the range 30μ to several millimeters in diameter, though not limited to planktonic organisms. Cowen and Guigand, 2008 employ continuous flow of seawater through a specialized chamber, using a digital camera to sample plankton. The device is shown in Figure 3.1e.



(a) The schematics of AILARON.



(b) A depiction of AILARON.



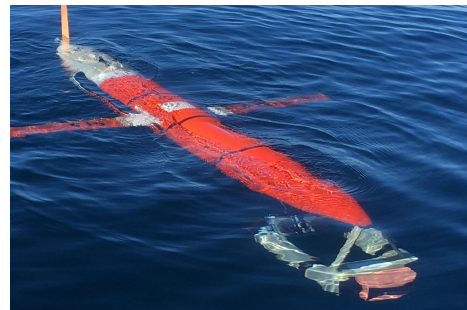
(c) Continuous Plankton Recorder (CPR).



(d) The In situ Ichthyoplankton Imaging System (ISIIS).



(e) FlowCytobot.



(f) Zooglider.

Figure 3.1: Underwater devices for plankton image sampling. (a) and (b): AILARON, a 2.35 meters long and equipped with in-situ sensors, including a Silhouette Camera for image sampling Source: Saad et al., 2020. (c): CPR, a towed device for large-scale, long-distance ocean surveys. Water flows through the tip of the device shown on the left in the image. Source: *The Little Plankton Recorder That Could* n.d. (d): ISIIS, a towed device utilizing shadowgraph technique sampling. Source: *IFCB BBG* n.d. (e): FlowCytobot, a submersible instrument employing a continuous-flow chamber and a laser detection system. Source: *Facebook Plankton Image* n.d. (f): Zooglider, a UAV designed for in-depth zooplankton observation. Source: *flickr Plankton Image* n.d.

The presented approaches employ ship-based or long-term monitoring at discrete locations, often lacking spatial and temporal resolution. As an alternative, the utilization of AUVs has been proposed. Ohman et al., 2019 presents a UAV which follows a pre-set path, capturing high-resolution images and acoustic data of zooplankton. The UAV is depicted in Figure 3.1f. Finally, in Saad et al., 2020, AILARON is presented, a novel tool for in-situ sampling and classification of plankton. Depicted in Figure 3.1b, AILARON is a novel tool for in-situ sampling and classification of plankton. The UAV processes images in real-time to detect patterns indicating the presence of a plankton hotspot, such as rapid changes in the detected parameters or a specific combination of conditions.

3.2 Existing Plankton Datasets

This section describes the following plankton image datasets: the Kaggle National Data Science Bowl (NDSB) plankton dataset, the AILARON dataset, and the Woods Hole Oceanographic Institution (WHOI) dataset. Additional datasets are presented Pastore et al., 2020 and Davies et al., 2017. The gray-scale images in the NDSB dataset (Cowen and Guigand, 2008) were sampled in Florida using ISIS. A subset of 30,000 images, including several plankton size groups, was segmented and labeled into 121 classes. The images vary in size, with some being partial due to imperfect segmentation. Example images are presented in Figure 3.2. The AILARON dataset is composed of marine plankton sampled in the Trondheim Fjord. The dataset comprises 3,619 labeled RGB images, divided into 21 distinctive classes. These classes encompass a variety of plankton taxa and other marine entities including, but not limited to, bubbles, cnidaria, coscinodiscus, echinodermata, jellyfish, and crustacea, as well as more general categories such as eggs, fecal pellets, and diverse forms of matter, indicated as "other". Example images are presented in Figure 3.3. The exact size range is not specified. The WHOI-Plankton dataset (Sosik, Peacock et al., 2015), collected by the Woods Hole Oceanographic Institution, consists of over 3.4 million expert-labeled plankton images spanning 70 classes. Gathered using an in-situ Imaging FlowCytobot at Martha's Vineyard Coastal Observatory, the dataset includes samples from different plankton size groups.

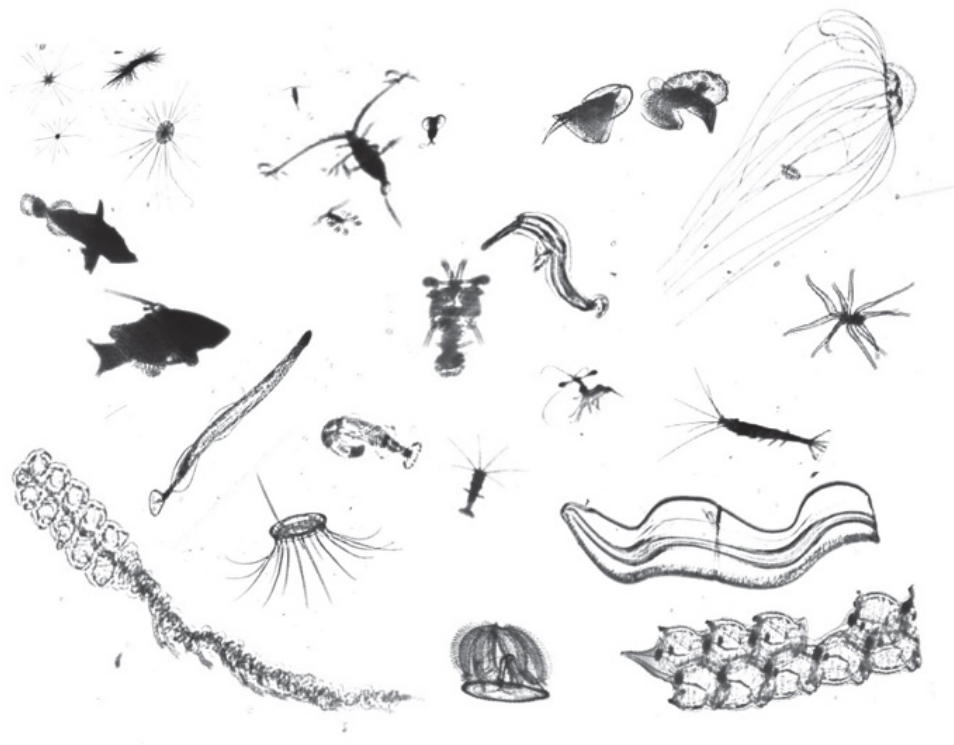


Figure 3.2: Example of the gray-scale images from the NDSB dataset. Containing 121 classes, the dataset consists of a wide range of plankton size groups. Source: Kaggle, 2023.

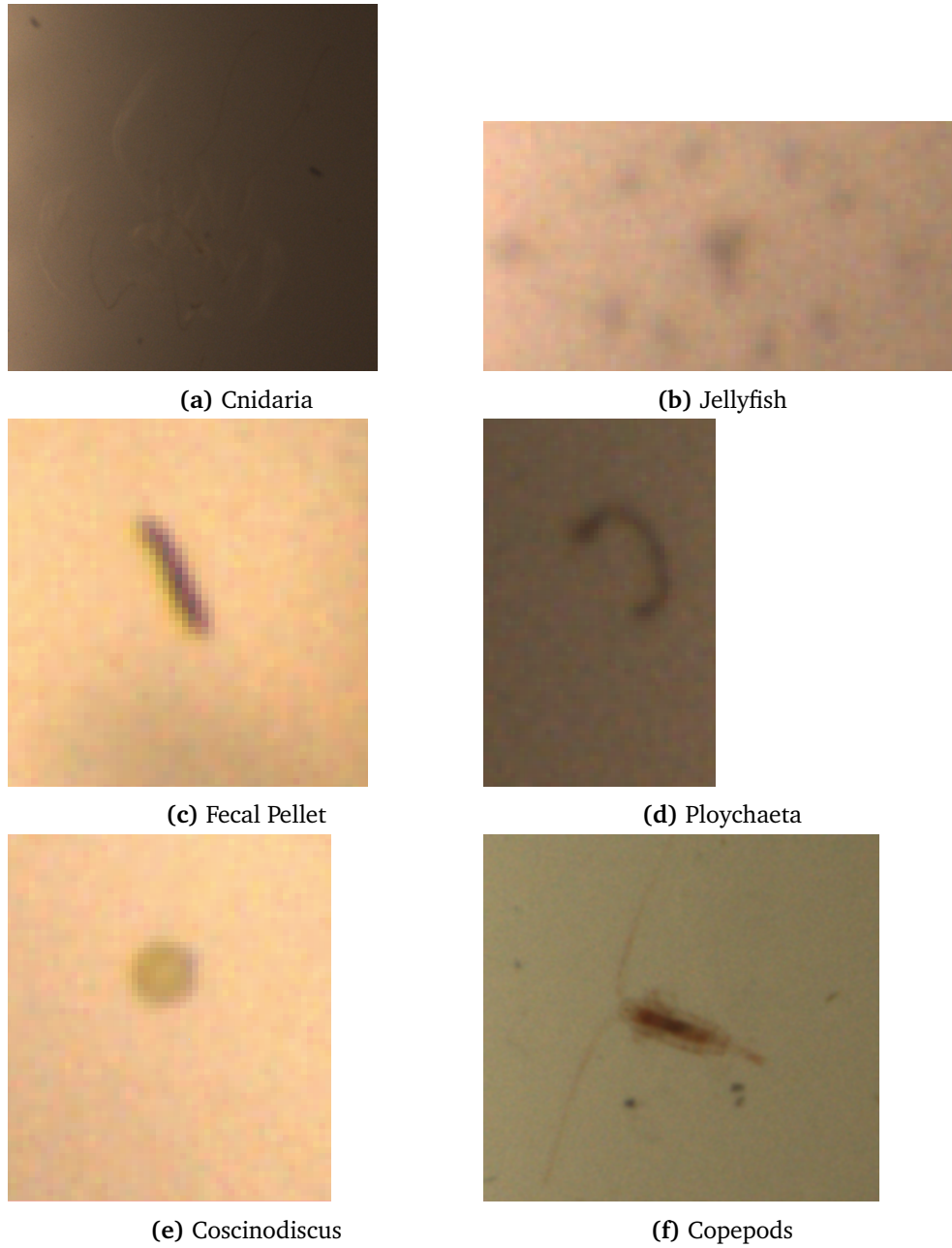


Figure 3.3: The figure shows multiple samples from the AILARON datasets. The images are of different resolutions and illumination depending on their position relative to the Silhouette Camera during sampling.

3.3 Plankton Classification

In recent years, the labor-intensive process of manually labeling plankton samples has increasingly been complemented by automated approaches using machine learning. This section describes the role of supervised and unsupervised machine learning in plankton image classification.

Supervised Classification

Supervised machine learning has proven effective in computer vision tasks such as object detection, segmentation, and classification. Consequently, this approach has been applied for plankton image classification. In Sosik and Olson, 2007, a Support Vector Machine (SVM) framework is proposed for classifying microplankton and smaller organisms. Their approach uses image analysis techniques to extract various features from these images, including size, shape, symmetry, texture characteristics, et al. Some of these features required preprocessing with techniques such as edge detection. The 22-category SVM classifier achieves an overall accuracy of 88% for an independent test set, with individual category accuracies ranging from 68% to 99%. However, The study states that the relevant identifying characteristics in the dataset must be sufficiently broad for the classifier to be effective. Moreover, the study states the necessity for adding new categories to the dataset.

Conversely, in Ohman et al., 2019 and Ellen, 2018, a convolutional neural network is applied for classification with no feature selection and extraction procedure. The study concludes that the proposed supervised methods were effective and that the dataset size had a greater impact on performance than the choice of ML model. The choice of hyperparameters had a considerable impact on performance.

Supervised methods have been applied for AILARON. Saad et al., 2020 propose a Deep Learning (DL) model for classification. The classifier achieved an accuracy of 95% on the seven-class dataset presented in Davies et al., 2017. The study acknowledges the insufficiency of the size and diversity of the dataset as a limitation of the study.

Unsupervised Classification

Supervised classification approaches rely on annotated datasets, imposing manual effort on taxonomists. Moreover, Culverhouse et al., 2003 presents evidence of the challenges experienced by human taxonomists and ecologists in accurately identifying marine dinoflagellates (a type of plankton), concluding that Trained personnel achieved 67-83% self-consistency and only 43% consensus with other experts in taxonomic labeling tasks. In contrast, experts with experience in specific discriminations demonstrated an 84-95% accuracy. The paper also suggests that automation methods can perform as well as humans in this categorization.

To address the challenges of labeling, active learning (AL) methods have been applied to select and label the most informative samples from a dataset. Haug

et al., 2021a¹ demonstrates how AL can minimize labeling efforts while attaining model performance. Haug et al., 2021b¹ extends this work, improving model performance by including data augmentation to improve model performance.

To circumvent the need for an annotated dataset, unsupervised approaches have been proposed. Salvesen et al., 2022 propose a framework consisting of data pre-processing, deep learning feature extraction, and classification. Autoencoder and Deep Cluster are considered for feature extraction (or representation learning), while k-means, Spectral Clustering (SC), Gaussian-mixture, and BIRCH are applied for clustering. The Autoencoder combined with SC obtains an accuracy of 14% on the full NDSB dataset. The study reveals that a rotational invariant unsupervised autoencoder effectively improves classifier performance. The study’s limitations include the time-consuming nature of the spectral clustering algorithm, making it less ideal for real-time applications.

Similarly, Salvesen et al., 2020 employs autoencoders and cluster algorithms for classification, achieving an accuracy of 76% on a five-class subset of the NDSB dataset. The results were considered insufficient compared to supervised approaches. There was also the need to extend the framework to automatically decide the number of initial clusters.

3.4 Mutual Information Maximization

In Linsker, 1988, R. Linsker describes how perceptual systems can self-organize to recognize specific features in their environment. The author draws parallels between the learning patterns of biological systems and the principles of mutual information maximization in unsupervised machine learning (coined as the Infomax Principal). For unsupervised learning, in the absence of labels, the choice of the loss function² is typically based on the structure and distribution of the data. In the following, we present a group of methods that propose a loss function, inspired by the Infomax Principle, based on maximizing mutual information.

Regularized Information Maximization (RIM)

Regularized Information Maximization (RIM), presented in Krause et al., 2010, proposes a loss function for learning a classifier by maximizing the mutual information between the input data and their label distribution. The method considers three aspects of the classifier, though only two are considered here, namely: class separation and classifier complexity. Class separation refers to the degree to which classes are distinguishable from each other, while classifier complexity refers to the simplicity of the model, with simpler models generally favored to avoid overfitting. Formally, given the random variables \mathbf{x} and \mathbf{z} representing the input data and their label distribution, respectively, RIM propose the loss function:

¹ This study is a part of AILARON

² The terms *loss function*, *optimization criterion* and *objective function* refers to the same concept, and are often used interchangeably.

$$\mathcal{R}(\theta) - \lambda I(\mathbf{x}; \mathbf{z}). \quad (3.1)$$

θ denotes the model parameters, $\mathcal{R}(\theta)$ is a regularization penalty, λ is a application specific trade-off parameter and $I(\mathbf{x}; \mathbf{z})$ denotes the mutual information between the two variables. The regularization term enforces classifier complexity, while the mutual information terms enforce class separation. The study demonstrates how RIM is able to outperform CS and maximum margin clustering (MMC).

Information Maximizing Self-Augmented Training (IMSAT)

Similar to RIM, *Information Maximizing Self-Augmented Training* (IMSAT) (Hu et al., 2017) proposes a loss function for maximizing the mutual information between the input and output of the classifier. IMSAT proposes a regularization term for imposing specific invariances on the learned label distribution, leveraging data augmentation to model these invariances. In simple terms, the method encourages the predicted label distribution of the augmented input to align closely with the predicted label distribution of the original input. The method reports a performance of 98.4% on the MNIST dataset. The study underlines that the performance heavily depends on the choice of data augmentation strategies and that these should be chosen to reflect the invariances required by the specific application.

Invariant Information Clustering (IIC)

In contrast to RIM and IMSAT, *Invariant Information Clustering* (IIC) (Ji et al., 2019) propose a loss function for maximizing mutual information between the function's classifications for paired data samples. The paired data samples are obtained by data augmentation. The entropy maximization component in mutual information ensures that the loss is not minimized if all images are assigned to the same class, making IIC effective in avoiding the problem of clustering degeneracy (a situation where one or a few clusters dominate the predictions). Formerly, given the predicted label distribution of a set of input data \mathbf{z} , and the label distribution of their augmentation \mathbf{z}' , IIC proposes the loss function:

$$-I(\mathbf{z}; \mathbf{z}'). \quad (3.2)$$

IIC archives promising results on several benchmark datasets, including an 88.8% accuracy on the STL10 dataset.

Chapter 4

Methodology

This chapter outlines the methodology for addressing the research questions in Section 1.1.1. The main objective of this thesis is to assess the effectiveness of a group of unsupervised methods for training a plankton image classifier. Accordingly, these methods will be introduced in Section 4.2. The applied datasets are described in Section 4.1, and a set of machine learning models to be used as classifiers are presented in Section 4.2.1.

4.1 Dataset Selection and Preprocessing

This thesis leverages two plankton datasets: the National Data Science Bowl (NDSB) and AILARON datasets. The training dataset of a classifier significantly impacts its performance and ability to generalize to unseen data. Aspects to consider include the number of training instances and representativeness (reflecting the diversity for the real-world application) of the dataset. The AILARON dataset is insufficient in both these aspects. The NDSB dataset will be applied to train and assess the proposed methods due to its adequate size and diversity (or AILARONs lack thereof). This dataset will serve as a comparative reference to AILARON. Given the obtained results and the resemblance between the two datasets, we can infer the suitability of the proposed method for AILARON. Additionally, the NDSB dataset serves as a benchmark for comparing the proposed methods to Salvesen et al., 2022. The NDSB dataset will be applied in two cases: Case 1 incorporates all classes for benchmarking purposes, and Case 2 will consider a subset of ten distinct classes to provide more interpretable results.

For dataset preprocessing, images will be rescaled to handle aspect ratio variations. To exploit low-scale image features, smaller images will be upscaled. No additional preprocessing steps will be applied to keep application-specific tuning to a minimum.

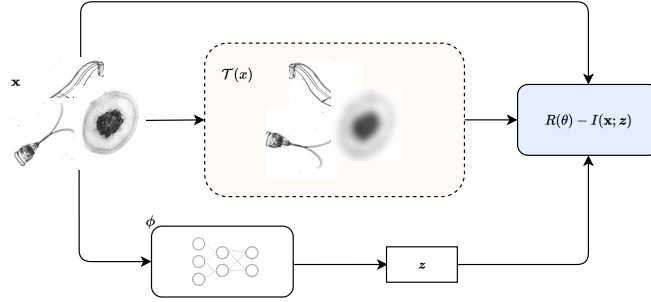


Figure 4.1: Regularized information Maximization (RIM): a loss function for learning a classifier by maximizing the mutual information between the input data and their label distribution. The classifier is modeled as a CNN. The augmentation function $\mathcal{T}(x)$ is chosen to impose the desired invariance.

4.2 Loss Function

This thesis proposes three methods, or loss functions, for learning a plankton image classifier: Regularized information Maximization (RIM), Information Maximizing Self-Augmented Training (IMSAT), and Invariant Information Clustering (IIC). These methods are tasked with the following: given a dataset $\mathcal{D} = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ of N sample images, we wish to derive a classifier $\phi : \mathcal{X} \rightarrow \mathcal{Z}$ that maps \mathbf{x}_n to a C -dimensional probability vector $\mathbf{z} \in \mathcal{Z} = [0, 1]^C$. The element z_c in \mathbf{z} represents the probability that the sample \mathbf{x}_n belongs to the c 'th class.

Regularized information Maximization (RIM)

RIM, illustrated in Figure 4.1, minimizes the objective defined in Equation (3.2). Inspired by Hu et al., 2017, this thesis proposes a regularization term that applies image augmentation techniques to impose intended invariances on the label distributions. The representation invariance is achieved by penalizing the dissimilarities between the original and augmented image representation. The applied image augmentation is specified by the augmentation function $\mathbf{x}' = \mathcal{T}(\mathbf{x})$. Consider the label distributions of a image-augmentation pair, $\mathbf{z} = \phi(\mathbf{x})$ and $\mathbf{z}' = \phi(\mathbf{x}')$ respectively. The similarity, or dissimilarity, between these two distributions can be quantified by the Kullback-Leibler (KL) divergence. The invariance is enforced by adjusting the classifier parameter weights θ so that the KL divergence $D_{\text{KL}}(\mathbf{z} | \mathbf{z}')$ is minimized. The mutual information term remains as presented in Equation (3.2). The proposed loss function is:

$$\mathcal{R}(\theta; \mathbf{z}, \mathbf{z}') - I(\mathbf{x}; \mathbf{z}), \quad \text{where, } \mathcal{R}(\theta; \mathbf{z}, \mathbf{z}') = D_{\text{KL}}(\mathbf{z} | \mathbf{z}').$$

The applied augmentation includes random image rotation, reflections, scaling, Gaussian blurring, random sharpness, and brightness jittering.

Information Maximizing Self-Augmented Training (IMSAT)

Information Maximization Self-Augmented Training (IMSAT) expands on the RIM method by introducing a local perturbation-based augmentation function, $\mathcal{T}(\mathbf{x}) = \mathbf{x} + r$. As stated in Hu et al., 2017: "The use of local perturbation ... encourages the data representation to be locally invariant". The applied perturbation is based on *Virtual Adversarial Training* as described by Miyato et al., 2015:

$$r = \arg \max_r \{ \mathcal{R}(\theta; \mathbf{x}, \mathbf{x}'); \|r\|_2 \leq \epsilon \} \quad \text{where } \mathbf{x}' = \mathcal{T}(\mathbf{x}).$$

ϵ defines the range of the perturbation and serves as a hyperparameter. The solution for r is approximated by a pair of forward and backward passes according to Miyato et al., 2015. The mutual information term remains as presented in Equation (3.2), while the regularization term is defined according to Hu et al., 2017:

$$\mathcal{R}(\mathbf{x}, \mathbf{x}') = -\frac{1}{N} \cdot \sum_{n=1}^N \sum_{c=1}^C p(z_c | \mathbf{x}_n) \cdot \log p(z_c | \mathbf{x}'_n). \quad (4.1)$$

The form of equation A.1 is adapted for image classification tasks and deviates from the original formulation. Please refer to Appendix A.4 for a complete derivation.

Invariant Information Clustering (IIC)

Invariant Information Clustering (IIC) introduces a loss function that maximizes the mutual information between the label distributions of an image and its corresponding augmented version. Figure 4.2 illustrates this approach. According to Ji et al., 2019, for the label distribution \mathbf{z} and \mathbf{z}' , their conditional joint distribution, i.e., the probability of assigning \mathbf{x} to class c and \mathbf{x}' to class c' , can be formulated as $P(z = c, z' = c' | \mathbf{x}, \mathbf{x}') = \phi_c(\mathbf{x}) \cdot \phi_{c'}(\mathbf{x}')$. This formulation implies that the label distributions are conditionally independent, given their respective inputs. After evaluating all possible image-augmentation pairs in the dataset (or a batch), the joint probability distribution is given by the $C \times C$ matrix \mathbf{P} . The loss function is obtained by inserting \mathbf{P} in the expression for mutual information, Equation (2.4). The proposed loss function then becomes:

$$I(\mathbf{z}; \mathbf{z}') = \sum_{c=1}^C \sum_{c'=1}^C \mathbf{P}_{cc'} \cdot \ln \frac{\mathbf{P}_{cc'}}{\mathbf{P}_c \cdot \mathbf{P}_{c'}}. \quad (4.2)$$

\mathbf{P}_c and $\mathbf{P}_{c'}$ are the marginal probabilities of an image being assigned to class c and c' . Please refer to Appendix A.4 for a complete derivation. The applied augmentation is the same as for RIM.

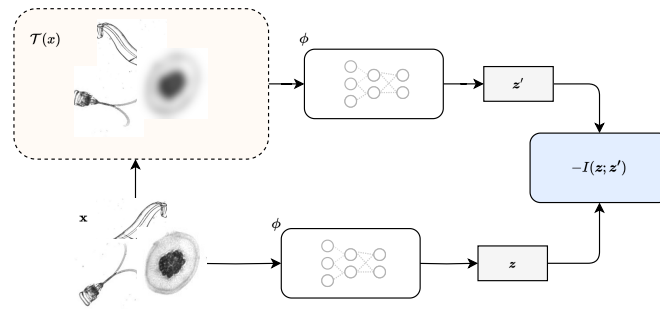


Figure 4.2: Invariant Information Clustering (IIC): a loss function for learning a classifier by maximizing the mutual information between the label distributions of an image-augmentation pair. The classifier is modeled as a CNN. The augmentation function $\mathcal{T}(x)$ is chosen to impose the desired invariance.

4.2.1 Machine Learning Models

The choice of model architecture in machine learning is not straightforward and must be determined based on the specific application. This thesis will consider a set of CNN models commonly used in computer vision tasks.

Residual Neural Network (ResNet)

The Residual Neural Network (ResNet), described in He et al., 2016, introduces the concept of "skip" or "shortcut" connections. These connections enable the model to bypass one or more layers by directly adding the output from the previous layer to the output of the skipped layers. The concept was introduced to address the challenge of vanishing gradients, a common issue in deep neural networks. This issue often leads to slower convergence and poorer performance. This architecture has achieved state-of-the-art results on various benchmarks.

Visual Geometry Group (VGG)

The VGG network, developed by the Visual Geometry Group, applied small 3x3 convolutional filters and the stacking of these filters to build deeper networks. Prior to VGG, larger filters (e.g., 5x5 or 7x7) were more commonly used. VGG demonstrated that multiple stacked smaller kernel convolution layers can mimic the receptive field of larger kernels but with the advantage of fewer parameters and lower computational complexity (Simonyan and Zisserman, 2014).

Densely Connected Convolutional Network (DenseNet)

The Densely Connected Convolutional Network, or DenseNet, introduces a CNN structure where each layer is directly connected to every subsequent layer. These dense connections create a highly integrated network where layers can directly

leverage the raw features from previous layers and pass on the learned features to all future layers. This architecture significantly reduces the number of parameters, making the network more computationally efficient and easier to train Huang et al., 2017.

Inception

Inception, or GoogLeNet, employs an approach where convolutions of varying sizes (1x1, 3x3, 5x5) are performed simultaneously within the same layer, and their outputs are concatenated. This "Inception Module" design enables the network to learn features at different spatial scales within the same layer. Further, 1x1 convolutions are used to reduce dimensionality before expensive 3x3 and 5x5 convolutions, improving computational efficiency. This strategy allows the network to learn multi-level features at each layer, making it more expressive Szegedy et al., 2015.

4.3 Evaluation

In this study, we will evaluate the unsupervised learning methods in two scenarios, one employing the labels of the dataset, and one considering the fully unsupervised scenario without labels. In the context of unsupervised classifiers, true labels are typically not used during the training phase. However, when evaluating the performance of an unsupervised classifier, it is possible to use true labels as a basis for validation. In this scenario, the assigned labels of the unsupervised classifier are arbitrary to the groups or clusters and do not necessarily align with the true labels. Therefore, a relabeling or matching procedure becomes necessary to align the labels assigned by the classifier with the true labels. One effective method for performing this relabeling is the Hungarian algorithm. In the context of evaluating unsupervised classifiers, it can be used to find the optimal one-to-one correspondence between the cluster labels assigned by the classifier and the true labels, such that some cost (e.g., the total number of misclassifications) is minimized. This relabeling procedure typically involves matching each cluster to the true label that is most prevalent within that cluster. For instance, if cluster A contains mostly samples of class 1, the label of cluster A will be changed to 1. This process ensures that the labels assigned by the classifier are in accordance with the true labels, enabling the computation of the accuracy metric.

In the second scenario, we will consider Normalized Mutual Information (NMI) as an unsupervised performance metric. NMI (Vinh et al., 2009) is widely used to assess unsupervised learning algorithms. This method quantifies the degree of correspondence between a classifier input and its corresponding label distribution. Formally, given the classifier input x and its assigned label distribution y , the NMI is obtained by:

$$\text{NMI}(\mathbf{x}, \mathbf{y}) = \frac{I(\mathbf{x}, \mathbf{y})}{\sqrt{H(\mathbf{x})H(\mathbf{y})}} \quad (4.3)$$

Mutual information and entropy, $I(\cdot)$ and $H(\cdot)$, are defined according to Equation (2.4) and Equation (2.1) respectively. A high entropy indicates a better performance. The correspondence between accuracy and NMI will be decisive for whether NMI is a suitable performance metric in the unsupervised case (i.e. when no labels are used for evaluation).

Chapter 5

Implementation

This chapter describes the implementation of the methodology presented in the previous chapter. Section 5.1 details the hardware and software setup of the study. The implementation of the dataset preprocessing steps is described in 5.2, while the implementation of the proposed loss functions and classifier models is described in 5.3 and 5.4, respectively. Section 5.5 provides a framework for training and evaluating the proposed methods. The implementation is accessible through Kwizera, 2023.

5.1 Hardware and Software Setup

The applied programming language for this study was Python(Python 3.9.12). The language was chosen for its ease of use and mature data science libraries. The libraries used for development are summarized in Table 5.1. The computational unit applied for this study was IDUN, a High Performance Computing (HPC) cluster at the Norwegian University of Science and Technology. IDUN consists of various computational resources. This study utilized the NVIDIA A100 Tensor Core GPU with 80G (gigabytes) of random access memory (RAM).

5.2 Dataset Preprocessing

The NDSB dataset was processed by resizing each image to a standard aspect ratio of 299 by 299. For RIM and IIC, the `torchvision.transforms` module from PyTorch was used to apply various image transformations. The specific transformations applied were `RandomHorizontalFlip`, `RandomRotation`, `RandomResizedCrop`, `ColorJitter`, and `GaussianBlur`. These transformations were executed for each image in the dataset, resulting in an augmentation pair. To ensure that the final classifier would be invariant to the orientation of an image, the symmetry transformation (horizontal flip) was applied with a probability of 50 %, and the rotation transformation with a random angle. Hence, all orientations of the transformed image were equally probable. The `ColorJitter` method was used for random

Library	Package	Description
PyTorch	torch	A machine learning library. Common applications are natural language processing and artificial intelligence.
scikit-learn	sklearn	A machine learning library. Features various classification, regression, and clustering algorithms and is designed to interoperate with the libraries NumPy and SciPy.
SciPy	scipy	A library built on NumPy for scientific and technical computing. It offers modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers, and more.
Python Imaging Library	PIL(Pillow)	A library for opening, manipulating, and saving many different image file formats. Extensively used in image processing tasks and supports a wide variety of images such as "jpeg," "png," "bmp," etc.

Table 5.1: The table shows the Python libraries that were used in the study.

brightness and sharpness adjustments. The crop implemented by the `RandomResizedCrop` resulted in a random area and aspect ratio. Gaussian blurring was applied using the `GaussianBlur` method.

5.3 Loss Function

This section describes the implementation of the proposed loss functions.

Information Maximizing Self-Augmented Training (IMSAT)

The IMSAT method, detailed in Algorithm 2 and 1, was implemented according to Hu et al., 2017 and Miyato et al., 2015. Algorithm 1 describes the implementation of self-augmented training (SAT) by virtual adversarial perturbation (VAT). This implementation approximates the adversarial perturbation d using the power iteration method Golub and Van der Vorst, 2000 and the finite difference method. The adversarial perturbation d is first initialized as a random unit vector. This perturbation is then added to the input data on line 4. Line 5 calculates the gradient of the KL divergence between the original output and the perturbed output w.r.t the perturbation, resulting in the direction in which a small change in the input

will most increase the KL divergence. d is then normalized. After calculating the adversarial direction, the perturbation is applied to the input data on line 8, and the final loss of the regularization term is calculated on line 10. The parameters ξ and ε are the constant for finite difference approximation and step size for the adversarial perturbation, respectively. Reported values for these parameters were used to keep application-specific tuning to a minimum.

Algorithm 1 Self-Augmented Training by Virtual Adversarial Perturbation

```

1: procedure SAT( $\phi, x, z, \varepsilon, \xi, I_p$ )
2:    $d \leftarrow$  Initialization by random unit vector
3:   for  $i \in \{1 \dots I_p\}$  do
4:      $z_p \leftarrow \phi(x + \xi d)$  ▷ Apply perturbation
5:      $d \leftarrow \nabla_{\xi d} D_{KL}(z, z')$  ▷ Calculate adversarial direction
6:      $d \leftarrow$  Normalized  $d$ 
7:   end for
8:    $z_p \leftarrow \phi(x + \varepsilon d)$ 
9:    $R \leftarrow D_{KL}(z_p, z)$ 
10:  return  $R$ 
11: end procedure

```

Algorithm 2 outlines the remaining IMSAT procedure. Similar to algorithm 3, the procedure is executed in batches. The method aims to learn a probabilistic classifier that maps similar inputs into similar outputs or discrete representations. Considering the input and output as the stochastic variables x and z , the classifier can be described as the conditional distribution $P(z|x)$. In simple terms, the distribution describes the probability of a class assignment given some input data. The conditional distribution is defined and set to the classifier’s output on line 2. The marginal distribution of z is calculated by approximation on line 3. For the explicit approximation, see Appendix A.4. We introduce the parameter \mathcal{B} denoting the batch size. The mutual information between the input and output is calculated on line 4 according to equation 2.4. The regularization term is then calculated according to algorithm 1. The final loss is calculated on line 7 weighted sum of the regularization term and the negative of the mutual information. The weighting of the mutual information term is defined by λ , which serves as a tuning parameter.

Regularized Information Maximization

The proposed variation of RIM can be implemented by applying minor changes to Algorithm 2. As described in Section 4.2, the proposed regularization term minimizes the KL divergence between augmentation pairs. This method is implemented by substituting line 5 in Algorithm 2 with $D_{KL}(z', z)$ where (z', z) denotes a batch

Algorithm 2 Regularized Information Maximization

```

1: procedure IMSAT( $\phi, x, z, \lambda, \varepsilon, \xi, I_p, \mathcal{B}$ )
2:    $p_{z|x} \leftarrow z$ 
3:    $p_z \leftarrow \frac{1}{B} \times \text{SUMCOLUMNS}(p_{z|x})$ 
4:    $I \leftarrow \text{MI}(p_{z|x}, p_z)$ 
5:    $R \leftarrow \text{SAT}(\phi, x, z, \varepsilon, \xi, I_p)$ 
6:    $loss \leftarrow R - \lambda \times I$ 
7:   return  $loss$ 
8: end procedure

```

of augmentation pairs.

Invariant Information Clustering (IIC)

The IIC method, detailed in Algorithm 3, was implemented according to Ji et al., 2019. The arrays z and z' represent the original-augmented image pair. The procedure was executed in batches so that z and z' contained a set of images, and the index of an image in z was the same as the index of its augmentation in z' . The procedure assumes prior knowledge of the number of classes C . In practice, this parameter can be determined using dedicated methods, e.g., the Elbow Method, Silhouette Analysis, et al. (Kodinariya, Makwana et al., 2013). The second line of code describes an entrywise summation of the outer product of each augmentation pair in the batch, resulting in the joint probability matrix. This matrix is symmetrized and normalized in lines 3 and 4. The marginal probability of assigning the original and transformed images to a given class is calculated on lines 6 and 7. The logarithm function is undefined for the input zero. Entries of the joint probability matrix smaller than a certain value are therefore set to ε . This parameter is a hardware-specific value representing the smallest defined float number. Finally, the loss is calculated according to Equation (A.2) on line 8.

5.4 Machine Learning Models

The ML model architectures were implemented using the `torchvision.models` module in PyTorch. This module provides various model architectures for computer vision and other applications. The applied models, along with their depth and number of trainable weights are summarized in **tbl:models**. The softmax function was applied to the output of the models to make the class assignments probabilistic.

Algorithm 3 Invariant Information Clustering (IIC) Loss Calculation

```

1: procedure IIC( $z, z', C, \varepsilon$ )
2:    $P \leftarrow \frac{1}{\text{LENGTH}(z)} \text{SUM}(z \times z'^T)$             $\triangleright$  Compute joint probability matrix
3:    $P \leftarrow \text{SYMMETRIZE}(P)$ 
4:    $P \leftarrow \text{NORMALIZE}(P)$ 
5:    $P \leftarrow \text{CLAMP}(P, \varepsilon)$ 
6:    $P_c \leftarrow \text{SUMROWS}(P)$                             $\triangleright$  Compute marginals
7:    $P_{c_t} \leftarrow \text{SUMCOLUMNS}(P)$ 
8:    $I \leftarrow P \times (\text{LOG}(P_c) + \text{LOG}(P_{c_t}) - \text{LOG}(P))$ 
9:   return  $I$ 
10: end procedure

```

Model	Number of layers	Number of parameters
resnet18	18	11,689,512
densenet121	121	7,978,856
inception_v3	48	27,161,264
vgg11	11	132,863,336

Table 5.2: The table summarizes the model architectures that were employed in the study.

5.5 Training and Evaluation

For case 1, initially, the NDSB dataset was randomly partitioned into training and test sets. The dataset was split such that 80% was assigned to the training set and 20% to the test set. Figure 5.1 illustrates this approach. As the figure shows, the labels of the training set were not utilized during classifier training. A slightly different approach was employed in case 2. Due to the small dataset size, the same split ratio was maintained while using k-fold cross-validation. No dedicated training set was set aside for testing. The training procedure involves the loss functions and classifier architectures detailed in Section 5.3 and Section 5.4, respectively. The `normalized_mutual_info_score` and `accuracy_score` methods from the `sklearn.metrics` module was utilized for the evaluation process. The approach was validated with the MNIST dataset, a comprehensive database of handwritten digits widely used in machine learning. A benchmarking exercise with this dataset affirmed the efficacy of our approach.

Algorithm 4 outlines the training procedure for the classifier. This algorithm is specific to IIC. However, minor changes are required to implement RIM and IMSAT: for RIM, a function call must replace line 7; for IMSAT, a new function call must replace lines 5, 6, and 7. The symbol \mathcal{E} specifies the number of training epochs,

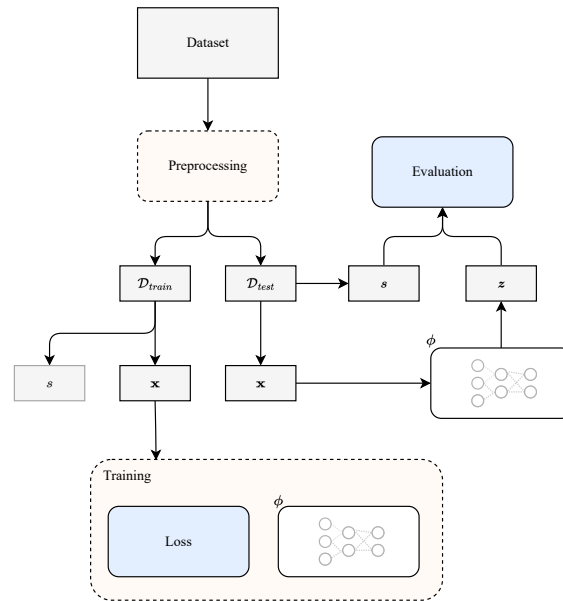


Figure 5.1: Diagram of the machine learning and evaluation process for one cross-validation split. The dataset is initially divided into a test set and a training set. Note that the labels of the training set are not used for classifier training.

and the batch size is denoted by \mathcal{B} . For RIM and IIC, the augmentation techniques described in section 5.2 are applied in line 5. Line 6 outlines a forward pass for the augmentation pair, while the loss is computed in line 7. The gradient of the loss function is calculated in line 8, and the model weights are updated in line 9.

In the UPDATEWEIGHTS procedure, the Adam optimizer from the `torch.optim` module was utilized. The batch size, chosen to be a power of two, was as large as allowed by the classifier models without exceeding memory constraints¹.

Given the absence of labels in unsupervised ML, determining the number of epochs by early stopping or adjusting the model’s hyperparameters by cross-validation is not straightforward. To minimize application-specific tuning, commonly reported hyperparameter values in the literature were used.

The classifier evaluation procedure was designed following Xie et al., 2016 and is outlined in algorithm 5. Initially, the arrays containing the true and predicted labels of the input data are set up as empty sets on lines 2 and 3. The trained classifier is then utilized on the input data on line 6, after which the true and predicted labels are appended to their respective arrays. To compute the accuracy of the unsupervised classifier on line 11, the predicted labels first have to be re-assigned using the HUNGARIANALGORITHM on line 10. To calculate the NMI score,

¹ The choice of batch size involves a trade-off between computational efficiency and the quality of the model’s statistical estimates. Batch sizes are often chosen as a power of two for GPU optimization.

Algorithm 4 Train machine learning model

```

1: procedure TRAIN( $D_{train}, \phi, \mathcal{E}, \mathcal{B}$ )
2:   for  $epoch \in \{1 \dots \mathcal{E}\}$  do
3:     for all  $batch$  in  $D_{train}$  split into  $\mathcal{B}$  do
4:        $x, s_b \leftarrow batch$ 
5:        $x' \leftarrow \text{AUGMENTINPUT}(x)$ 
6:        $x, x' \leftarrow \phi(x), \phi(x')$ 
7:        $loss \leftarrow \text{IIC}(y, y_t)$ 
8:        $g_w \leftarrow \nabla_w loss$ 
9:        $\phi \leftarrow \text{UPDATEWEIGHTS}(\phi, g_w)$ 
10:    end for
11:  end for
12: end procedure

```

the `normalized_mutual_info_score` was applied using the true labels and the classifier output as input parameters.

Algorithm 5 Test machine learning model

```

1: procedure UNSUPERVISEDCLUSTERINGACCURACY( $D_{test}, \phi, epochs, batch\_size$ )
2:    $z \leftarrow \emptyset$ 
3:    $s \leftarrow \emptyset$ 
4:   for all  $batch$  in  $D_{test}$  split into  $\mathcal{B}$  do
5:      $x, s_b \leftarrow batch$ 
6:      $z_b \leftarrow \phi(x)$ 
7:      $z \leftarrow \text{APPEND}(z, z_b)$ 
8:      $s \leftarrow \text{APPEND}(s, s_b)$ 
9:   end for
10:   $z \leftarrow \text{HUNGARIANALGORITHM}(s, z)$  ▷ Reassign the predicted labels.
11:   $acc \leftarrow \text{ACCURACY}(s, z)$ 
12:  return  $acc$ 
13: end procedure

```

Chapter 6

Results and Discussions

The methodology, outlined in Chapter 4, introduced a set of unsupervised methods based on mutual information maximization and image augmentation for plankton image classification. Furthermore, Section 4.2.1 explains that the choice of classifier architecture is not straightforward and should be specific to the application. This chapter discusses the results of applying these unsupervised methods in conjunction with the selected classifier architectures. Due to the inadequate size and diversity of the AILARON dataset, the study relies on the NDSB dataset for training and evaluation. An extensive examination of the NDSB dataset, its format, distribution, and how it compares to the AILARON dataset, is provided in Section 6.1. Lastly, Section 6.2 provides a detailed discussion of some of the limitations of the study.

6.1 Dataset Exploration and Preprocessing

The performance of a machine learning model heavily relies on the quality and diversity of the training set. On the publication site of the NDSB dataset (Kaggle, 2023), it is stated that "We made every effort to include a representation of real-world data in this dataset. In other words, we did not cherry-pick for the best and clearest images but used images spanning the gamut from blurry to clear and tiny to big". This variation in image format and plankton taxa presents several challenges. The density plot in Figure 6.1 illustrates the skewed distribution of the dataset. Please refer to Appendix B for the complete dataset distribution. The tail on the right side of the figure indicates that some few classes contain a considerable number of instances; the largest class includes 1979 instances, while the smallest contains only 9. Notably, the ten largest classes account for 37% of the dataset.

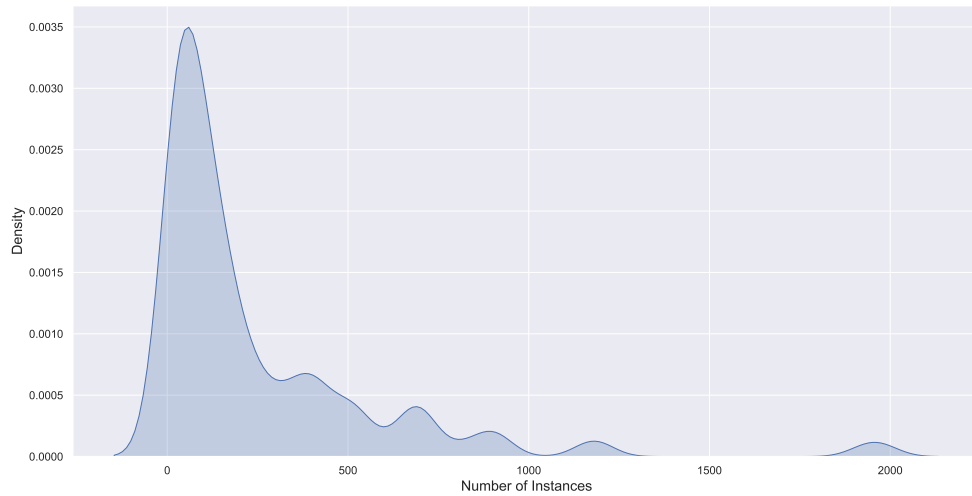


Figure 6.1: Density plot showing the NDSB dataset distribution. A small number of classes has a large number of class instances. The ten largest classes account for 37% of the dataset. Note that no classes have less than 9 instances. The curve spans the negative number line because of the smoothness characteristic of density plots.

A thorough inspection reveals substantial variations in image quality, similarities between instances of different classes, and high intra-class variability, as depicted in Figure 6.2. The presence of non-taxonomic distinctions in categories further complicates the classification task. For instance, some organisms of the same taxonomic group are classified by behavior, as in the case of the two classes, 'appendicularian_s_shape' and 'appendicularian_slight_curve', shown in Figure 6.2c and Section 6.1, respectively. In considering the AILARON dataset, as exemplified in Figure 3.3, it is clear there are significant differences in both image format and plankton distribution when compared to the NDSB dataset. As detailed in Section 3.2, the AILARON dataset encompasses only 21 classes. Additionally, these classes encapsulate a limited range of plankton organisms, while including marine particles represented by such classes as 'line', 'red stuff', and 'bubble'. Differences also exist in the format of images in the datasets. The NDSB dataset, captured using the ISIIS system, employs an in-line camera system that produces grayscale images (Cowen and Guigand, 2008). In contrast, the AILARON dataset utilizes a Silhouette Camera (SilCam), whose post-processing results in RGB images.

In considering a subset of the dataset for Case 2, ten distinct classes were selectively chosen, with their respective distributions listed in Table 6.1. These classes were distinctive in their appearance, offering no apparent similarities. The rationale for this selection was two-fold. First, it considerably simplified the challenges encountered in Case 1. Second, it provides a simple dataset for considering effective data augmentations techniques. By reducing the complexity and inherent ambiguity associated with similar classes (and variations within classes), the effects of data augmentation can become more apparent. Moreover, to mitigate

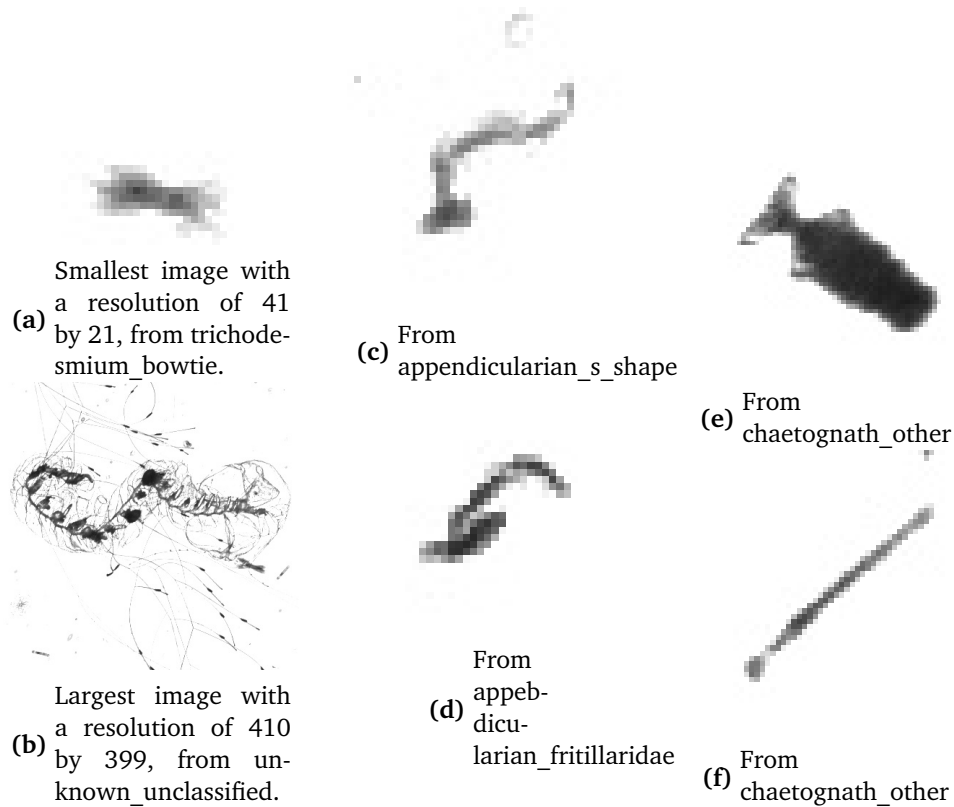


Figure 6.2: Examples from the NDSB dataset showing variations in image format, quality, and class variability. **(a):** The largest image in the dataset. **(b):** the smallest image in the dataset. **(c)** and **(d):** similarities between classes, non-taxonomic distinctions between plankton of the same group. **(e)** and **(f):** dissimilarities between instances of the same class.

the effect of an imbalanced dataset, the maximum number of instances per class was limited to 200, i.e. classes with 200 instances in Table 6.1 had additional instances in case 1. Note that these instances were chosen at random.

6.1.1 Unsupervised Classification

This section presents the results of the proposed unsupervised methods for two cases: one considering all classes of the NDSB dataset and another considering a subset of ten classes.

Case 1: All classes

Section 6.1.1 presents the performance of combinations of IMSAT and IIC with various classifier architectures, considering the entire dataset. The highest performance of 24.9% was achieved by combining the Inception architecture with the loss function proposed by IIC. The Inception and ResNet architectures, com-

Class	Number of Instances
acantharia_protist	200
chordate_type1	77
copepod_calanoid_eucalanus	96
copepod_cyclopoid_copilia	30
ctenophore_cestid	113
ctenophore_lobate	38
diatom_chain_string	200
echinoderm_larva_seastar_brachiolaria	200
hydromedusae_haliscera	200
radiolarian_chain	200

Table 6.1: The distribution of ten selective classes. The maximum number of instances per class was limited to 200 to mitigate the effects of an imbalanced dataset.

bined with IIC, outperformed the unsupervised method proposed in Salvesen et al., 2022 by 11% and 7%, respectively.

Classifier	Accuracy	
	IIC	IMSAT
VGG	7.3	8.9
DensNet	10.2	6.6
ResNet	21.1	11.6
Inception	24.9	8.4

Table 6.2: Classifier performance for the NDSB dataset. The inception architecture combined with IIC outperformance existing unsupervised methods for the specific application by 11%..

Case 2: Ten selective classes

Section 6.1.1 presents the performance of combinations of RIM, IMSAT, and IIC with various classifier architectures, for a subset of ten classes. The evaluation metrics include accuracy, balanced accuracy, and NMI. The best classifier, with an accuracy of 57.4%, was achieved by combining the DensNet architecture and IIC. Furthermore, the variation of RIM presented in this thesis outperforms IMSAT by 16.2%. The table highlights a strong relationship between supervised evaluation metrics, accuracy, and balanced accuracy, and the unsupervised metric, Normalized Mutual Information (NMI). These metrics provide consistent assessments of the classifiers' relative performance, despite some minor differences. It is important to note that the balanced accuracy of the classifiers is considerably lower than the absolute accuracy. This difference suggests that the proposed methods may

struggle with unbalanced datasets. More specifically, the classifiers effectively classify well-represented classes in the dataset, while they often misclassify underrepresented ones. The gap between accuracy and balanced accuracy is notably larger for the IIC model, up to 17%, whereas it hovers around 10% for both RIM and IMSAT. As presented in Ji et al., 2019, it is stated that IIC successfully avoids degenerate solutions (a situation where one cluster dominates the predictions). This characteristic might contribute to the comparatively higher performance of IIC. Yet, managing unbalanced datasets remains a challenge for this application.

Method	Model	Evaluation Metric		
		ACC	Balanced ACC	NMI
RIM	VGG	23.1(.01)	13.8(.00)	.099(.01)
	DensNet	38.2(.63)	30.6(0.04)	.345(0.08)
	ResNet	37.9(.10)	28.2(.08)	.314(.13)
	Inception	37.2(.05)	28.1(.08)	.296(.05)
IMSAT	VGG	21.0(.02)	12.7(.01)	.073(.04)
	DensNet	22.0(.01)	13.2(.00)	.091(.00)
	ResNet	18.1(.01)	10.4(.01)	.003(.01)
	Inception	21.2(.04)	12.8(.03)	.065(.06)
IIC	VGG	20.4(.02)	13.0(.08)	.123(.02)
	DensNet	57.4(.03)	40.1(.03)	.595(.03)
	ResNet	55.9(.05)	40.5(.05)	.569(.03)
	Inception	56.1(.06)	38.6(.05)	.553(.06)

Table 6.3: Classifier performance (%) and standard deviation for a subset of ten classes for the NDSB dataset. The table shows a strong correspondence between accuracy, balanced accuracy, and NMI as evaluation metrics. The highest performance was obtained by combining Inception and IIC. The proposed variation of RIM outperformed IMSAT by 16.2%.

The misclassification of underrepresented classes becomes more apparent when considering the confusion matrix in Figure 6.3. The matrix considers the DensNet classifier trained with IIC. This model effectively classifies the classes with 200 instances, which correspond to classes 0, 4, 6, 7, 8, and 9. However, the classification accuracy for the remaining classes is insufficient, reaching 0% in some cases. This situation represents a recurring challenge in unsupervised learning, as dataset-balancing techniques cannot be implemented in the absence of labels. The classifier achieves an accuracy ranging from 95% to 70% for the larger classes. Notably, IIC, as presented in Ji et al., 2019, asserts its robustness against clustering degeneracy, which refers to the tendency for a single cluster to dominate the predictions. This attribute might be a contributing factor to IIC's superior performance when compared to RIM and IMSAT.

Figure 6.4 illustrates the classifier's learning process for IMSAT and IIC. The relationship between the performance metrics (accuracy and NMI) is evident through-



Figure 6.3: Confusion Matrix for classifier of the Inception architecture trained with IIC. Classes 1, 2, 3, and 5 are misclassified and underrepresented in the dataset.

out training, indicating that NMI is sufficient as a performance metric in the unsupervised scenario. Note that the learning tends to plateau after about 50 epochs.

6.1.2 Image Augmentations

The results presented for RIM and IIC were achieved using the image transformations described in Section 5.2, with examples of augmentation pairs provided in Figure 6.5. Although numerous image transformations were tested, the following proved effective in enhancing model performance: Gaussian Blurring, brightness adjustments, sharpness adjustments, scaling, rotation, and reflection. Transformations producing plausible images similar to those in the original dataset are successful in improving model performance. This observation aligns with findings in existing literature (Perez and Wang, 2017 and Shorten and Khoshgoftaar, 2019).

6.2 Limitations of the Study

The results of the study were considerably influenced by hardware limitations, particularly in terms of memory and computation. Due to these limitations, certain results could not be obtained in Case 1, including the evaluation of RIM and the utilization of NMI as a performance metric. The computational constraints also impacted the hyperparameter tuning process. Due to limited computational resources and time, a more exhaustive hyperparameter search was not conducted, and widely reported values were used instead. While these values are a reasonable starting point, the sensitivity of the classifier’s performance to hyperparameters means that the reported results should be interpreted as indicative rather than

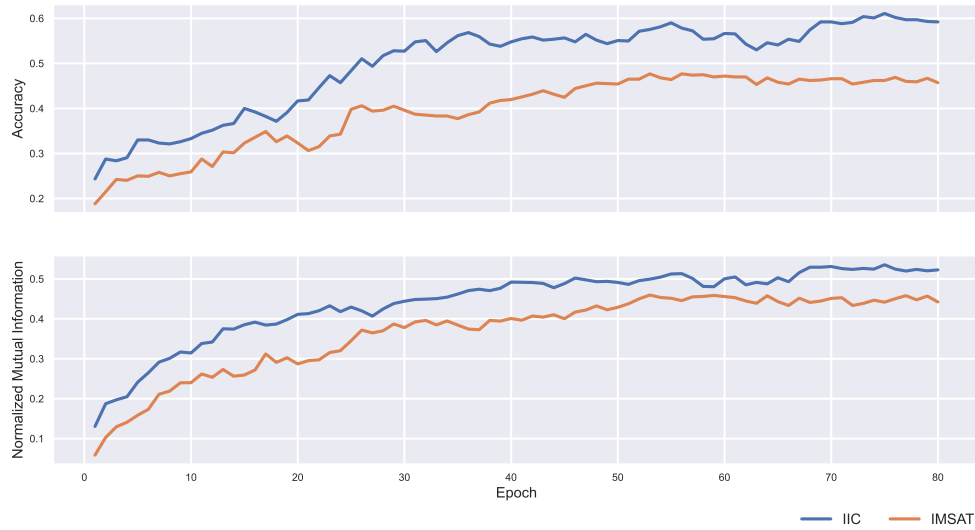


Figure 6.4: Classifier training using IIC(blue) and IMSAT(red). The plot indicates that 50 epochs are sufficient to obtain the optimal performance for the applied augmentations. IIC is superior throughout training.

definitive. Additionally, the preprocessing of input data introduced computational demands, especially with the high aspect ratio of 299 by 299 used for image rescaling. This aspect ratio was chosen to leverage low-level features but increased memory requirements. Considering that most images had lower resolutions, a more moderate aspect ratio could have been employed to increase batch size, reduce memory usage, and allow for a more comprehensive hyperparameter search. In conclusion, the study was affected by memory and computational constraints, impacting algorithm evaluation, performance metric utilization, hyperparameter exploration, and data preprocessing choices. These limitations should be taken into account when interpreting the results, and addressing these constraints could potentially lead to improved performance.

The study makes several pragmatic simplifications. Firstly, the number of classes is assumed to be known throughout this study. However, in the unsupervised setting, dedicated methods must be used to obtain this parameter. Furthermore, the thesis infers the applicability of data augmentation techniques by extrapolating their effectiveness from a subset of the NDSB dataset to the AILARON dataset. This is a gross assumption, but it is made necessary due to the limited size of the AILARON dataset. Another deviation from common practice is the methodology used in Case 2. Instead of employing a separate test set, the study only relies on k-fold cross-validation. The rationale for this was based on the small size of the dataset. It should be recognized that for small datasets, the performance of a model can fluctuate significantly between training and test sets. In summary, the thesis adopts known class numbers, makes an assumption about data augmentation based on a different dataset, and uses k-fold cross-validation instead of a

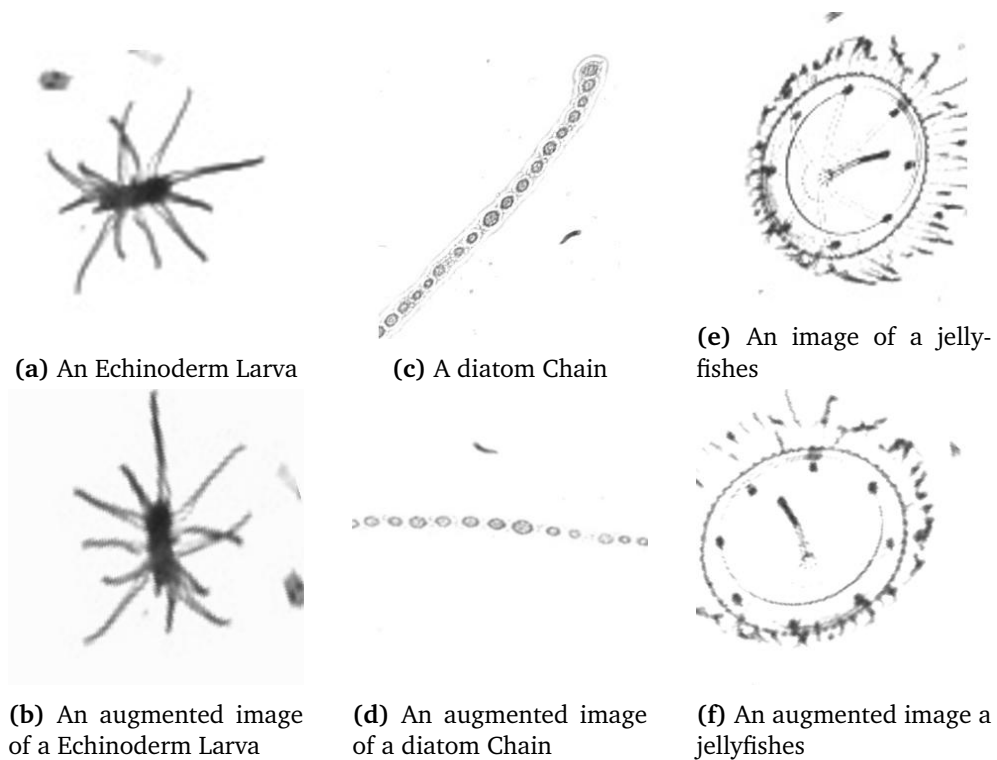


Figure 6.5: The figure shows examples of the applied image augmentations. The first row depicts images from the dataset, while the second row depicts their transformation. The transformations, Gaussian Blurring, brightness adjustments, sharpness adjustments, scaling, rotation, and reflection, were applied to each image to obtain an augmentation pair.

separate test set due to the small dataset size. These simplifications are aimed at practicality but should be taken into consideration when interpreting the results.

6.3 Applicability for AILARON

Given these substantial differences between the NDSB and AILARON datasets, inferring the applicability of unsupervised methods for the AILARON dataset based solely on performance with the NDSB dataset should be approached with caution. It is imperative to consider the differences in image format, class distribution, and classification criteria. Ideally, additional validation should be performed directly on the AILARON dataset to assess how well the unsupervised methods generalize to this specific dataset and its unique characteristics. Also, an additional step is required: post-classification, the clusters must be manually examined and labeled by taxonomists. Nonetheless, this study does demonstrate the potential of the proposed methods for AILARON. Furthermore, the proposed methods can serve as valuable tools for taxonomists, enabling an initial rough categorization of plankton images and streamlining the classification process.

Chapter 7

Conclusion

This thesis considers the application of unsupervised machine learning for in-situ plankton image classification. The thesis addresses the limitation of supervised approaches, including the need for extensive annotated datasets and the challenge of classifying unseen plankton. The thesis proposes a variation of Regularized Information Maximization (RIM), Information Maximizing Self-Augmented Training (IMSAT), and Invariant Information Clustering (IIC) for training a classifier in an unsupervised manner. The study demonstrates the potential of the proposed method for AILARON, surpassing previous unsupervised techniques by 11% by reaching an accuracy of 24.9% on the NDSB dataset. IIC proved to be the most effective method, followed by RIM and IMSAT. The comparison of the accuracy of the proposed methods with Normalized Mutual Information (NMI) revealed a strong relationship, confirming NMI as a reliable unsupervised performance metric. Moreover, the study experimented with multiple augmentation techniques, and a subset of these demonstrated notable effectiveness in enhancing classifier performance. Notably, these methods consistently produced plausible images within the dataset, aligning with existing literature findings.

While the results obtained on the NDSB dataset demonstrate promising performance, caution must be exercised in directly inferring the applicability of unsupervised methods to the AILARON dataset. The significant differences between the two datasets in terms of image format, class distribution, and classification criteria necessitate further validation directly on the AILARON dataset. Additionally, manual examination and labeling of the clusters by taxonomists are crucial steps to ensure accurate classification. Nevertheless, the proposed methods offer valuable tools for taxonomists, streamlining the initial categorization of plankton images and facilitating the overall classification process.

Chapter 8

Future Work

Though this thesis provides promising results, further research is required to obtain an effective approach for in-situ plankton classification for AILARON. The following outlines potential future directions.

- **AILARON Dataset Expansion:** The proposed methods require application-specific validation. However, the current dataset is limited in size, and diversity, and contains various marine entities besides plankton images. To enable a more comprehensive evaluation, a larger and more diverse AILARON dataset needs to be built. This entails annotating and labeling additional images.
- **Real-time Application:** The proposed method must be evaluated in a real-time in-situ classification setting. This involves optimizing the models for computational efficiency.
- **Hyperparameter Tuning:** Experimenting with various hyperparameters can enhance performance. Utilizing systematic search methods to identify optimal hyperparameter combinations.
- **Incorporation of Domain Knowledge:** Incorporating domain knowledge in the form of prior information about plankton characteristics could potentially enhance the quality of clustering. This might involve using hybrid approaches that combine unsupervised learning with domain-specific insights.

Bibliography

- Barton, Andrew D, Andrew J Pershing, Elena Litchman, Nicholas R Record, Kyle F Edwards, Zoe V Finkel, Thomas Kiørboe and Ben A Ward (2013). ‘The biogeography of marine plankton traits’. In: *Ecology letters* 16.4, pp. 522–534.
- Calanoida* (n.d.). <https://en.wikipedia.org/wiki/Calanoida>. Accessed: 2023-06-13.
- Chust, Guillem, Meike Vogt, Fabio Benedetti, Teofil Nakov, Sébastien Villéger, Anais Aubert, Sergio M Vallina, Damiano Righetti, Fabrice Not, Tristan Biard et al. (2017). ‘Mare incognitum: A glimpse into future plankton diversity and ecology research’. In: *Frontiers in Marine Science* 4, p. 68.
- Cover, Thomas M (1999). *Elements of information theory*. John Wiley & Sons.
- Cowen, Robert K and Cedric M Guigand (2008). ‘In situ ichthyoplankton imaging system (ISIIS): system design and preliminary results’. In: *Limnology and Oceanography: Methods* 6.2, pp. 126–132.
- Culverhouse, Phil F, Robert Williams, Beatriz Reguera, Vincent Herry and Sonsoles González-Gil (2003). ‘Do experts make mistakes? A comparison of human and machine identification of dinoflagellates’. In: *Marine ecology progress series* 247, pp. 17–25.
- Davies, Emlyn John, Per Johan Brandvik, Frode Leirvik and Raymond Nepstad (2017). ‘The use of wide-band transmittance imaging to size and classify suspended particulate matter in seawater’. In: *Marine pollution bulletin* 115.1-2, pp. 105–114.
- Dosovitskiy, Alexey, Jost Tobias Springenberg, Martin Riedmiller and Thomas Brox (2014). ‘Discriminative unsupervised feature learning with convolutional neural networks’. In: *Advances in neural information processing systems* 27.
- Ellen, Jeffrey Scott (2018). *Improving biological object classification in plankton images using convolutional neural networks, geometric features, and context metadata*. University of California, San Diego.
- Facebook Plankton Image (n.d.). https://scontent-arn2-1.xx.fbcdn.net/v/t39.30808-6/309032073_615998746887586_23169203441136839_n.jpg?_nc_cat=107&ccb=1-7&_nc_sid=09cbfe&_nc_ohc=8j3yKbX4WIMAX9PfyW0&_nc_ht=scontent-arn2-1.xx&oh=00_AfCN0McQDBqWYyBR2uhW-lfZ-Eb-qVzeAwvByXGpu.
- flickr Plankton Image (n.d.). <https://www.flickr.com/photos/scrippscocean/46576328121>.

- Gammarus roeselii* (n.d.). https://upload.wikimedia.org/wikipedia/commons/3/3f/Gammarus_roeselii.jpg. Accessed: 2023-06-13.
- Golub, Gene H and Henk A Van der Vorst (2000). 'Eigenvalue computation in the 20th century'. In: *Journal of Computational and Applied Mathematics* 123.1-2, pp. 35–65.
- Gonzalez, Rafael C (2009). *Digital image processing*. Pearson education india.
- Goodfellow, Ian, Yoshua Bengio and Aaron Courville (2016). *Deep learning*. MIT press.
- Hallegraeff, GM, DM Anderson and AD Cembella (2004). *Manual on harmful marine microalgae—monographs on oceanographic methodology*.
- Haug, Martin Lund, Aya Saad and Annette Stahl (2021a). 'A combined informative and representative active learning approach for plankton taxa labeling'. In: *Thirteenth International Conference on Digital Image Processing (ICDIP 2021)*. Vol. 11878. SPIE, pp. 495–503.
- Haug, Martin Lund, Aya Saad and Annette Stahl (2021b). 'CIRAL: a hybrid active learning framework for plankton taxa labeling'. In: *IFAC-PapersOnLine* 54.16, pp. 450–457.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren and Jian Sun (2016). 'Deep residual learning for image recognition'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.
- Hu, Weihua, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto and Masashi Sugiyama (2017). 'Learning discrete representations via information maximizing self-augmented training'. In: *International conference on machine learning*. PMLR, pp. 1558–1567.
- Huang, Gao, Zhuang Liu, Laurens Van Der Maaten and Kilian Q Weinberger (2017). 'Densely connected convolutional networks'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708.
- IFCB BBG (n.d.). https://mclanelabs.com/wp-content/uploads/2017/08/IFCB_BBG.jpg. Accessed: 2023-06-13.
- Ji, Xu, Joao F Henriques and Andrea Vedaldi (2019). 'Invariant information clustering for unsupervised image classification and segmentation'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9865–9874.
- Kaggle (2023). *Data Science Bowl*. Accessed: 2023-06-13. URL: <https://www.kaggle.com/competitions/datasciencebowl/data>.
- Kodinariya, Trupti M, Prashant R Makwana et al. (2013). 'Review on determining number of Cluster in K-Means Clustering'. In: *International Journal* 1.6, pp. 90–95.
- Krause, Andreas, Pietro Perona and Ryan Gomes (2010). 'Discriminative clustering by regularized information maximization'. In: *Advances in neural information processing systems* 23.
- Kullback, Solomon and Richard A Leibler (1951). 'On information and sufficiency'. In: *The annals of mathematical statistics* 22.1, pp. 79–86.

- Kwizera, Fred (2023). *Unsupervised Machine Learning for Plankton Classification*. Accessed: 10.6.23. URL: <https://github.com/Fredikw/ms-unsupervised-plankton-classification>.
- Lalli, Carol and Timothy Parsons (1997). *Biological oceanography: an introduction*. Elsevier.
- Lampert, W and U Sommer (2007). *The ecology of lakes and streams*.
- Lethal algae blooms an ecosystem out of balance* (n.d.). <https://www.theguardian.com/environment/2020/jan/04/lethal-algae-blooms-an-ecosystem-out-of-balance>. Accessed: 2023-06-13.
- Linsker, Ralph (1988). 'Self-organization in a perceptual network'. In: *Computer* 21.3, pp. 105–117.
- Makoto, Omori and Ikeda Tsutomu (1984). *Methods in marine zooplankton ecology*.
- Meroplankton* (n.d.). <https://australian.museum/learn/animals/plankton/meroplankton/>. Accessed: 2023-06-13.
- Mikrofoto.de-Raedertier-14* (n.d.). <https://upload.wikimedia.org/wikipedia/commons/e/ec/Mikrofoto.de-Raedertier-14.jpg>. Accessed: 2023-06-13.
- Miyato, Takeru, Shin-ichi Maeda, Masanori Koyama, Ken Nakae and Shin Ishii (2015). 'Distributional smoothing with virtual adversarial training'. In: *arXiv preprint arXiv:1507.00677*.
- Moltmann, Tim, Jon Turton, Huai-Min Zhang, Glenn Nolan, Carl Gouldman, Laura Griesbauer, Zdenka Willis, Ángel Muñoz Piniella, Sue Barrell, Erik Andersson et al. (2019). 'A global ocean observing system (GOOS), delivered through enhanced collaboration across regions, communities, and new technologies'. In: *Frontiers in Marine Science* 6, p. 291.
- Ohman, Mark D, Russ E Davis, Jeffrey T Sherman, Kyle R Grindley, Benjamin M Whitmore, Catherine F Nickels and Jeffrey S Ellen (2019). 'Zooglider: an autonomous vehicle for optical and acoustic sensing of zooplankton'. In: *Limnology and Oceanography: Methods* 17.1, pp. 69–86.
- Olson, Robert J and Heidi M Sosik (2007). 'A submersible imaging-in-flow instrument to analyze nano-and microplankton: Imaging FlowCytobot'. In: *Limnology and Oceanography: Methods* 5.6, pp. 195–203.
- Paerl, Hans W, Wayne S Gardner, Karl E Havens, Alan R Joyner, Mark J McCarthy, Silvia E Newell, Boqiang Qin and J Thad Scott (2016). 'Mitigating cyanobacterial harmful algal blooms in aquatic ecosystems impacted by climate change and anthropogenic nutrients'. In: *Harmful Algae* 54, pp. 213–222.
- Paerl, Hans W and Jef Huisman (2008). 'Blooms like it hot'. In: *Science* 320.5872, pp. 57–58.
- Paerl, Hans W and Jef Huisman (2009). 'Climate change: a catalyst for global expansion of harmful cyanobacterial blooms'. In: *Environmental microbiology reports* 1.1, pp. 27–37.
- Parsons, TR and CM Lalli (2002). 'Jellyfish population explosions: revisiting a hypothesis of possible causes'. In: *La mer* 40.3, pp. 111–121.

- Pastore, Vito P, Thomas G Zimmerman, Sujoy K Biswas and Simone Bianco (2020). 'Annotation-free learning of plankton for classification and anomaly detection'. In: *Scientific reports* 10.1, pp. 1–15.
- Perez, Luis and Jason Wang (2017). 'The effectiveness of data augmentation in image classification using deep learning'. In: *arXiv preprint arXiv:1712.04621*.
- Reid, Philip C, JM Colebrook, JBL Matthews, JCPR Aiken and Continuous Plankton Recorder Team (2003). 'The Continuous Plankton Recorder: concepts and history, from Plankton Indicator to undulating recorders'. In: *Progress in Oceanography* 58.2-4, pp. 117–173.
- Rost, Björn, Ingrid Zondervan and Dieter Wolf-Gladrow (2008). 'Sensitivity of phytoplankton to future changes in ocean carbonate chemistry: current knowledge, contradictions and research directions'. In: *Marine ecology progress series* 373, pp. 227–237.
- Saad, Aya, Annette Stahl, Andreas Våge, Emlyn Davies, Tor Nordam, Nicole Aberle, Martin Ludvigsen, Geir Johnsen, João Sousa and Kanna Rajan (2020). 'Advancing ocean observation with an ai-driven mobile robotic explorer'. In: *Oceanography* 33.3, pp. 50–59.
- Salps Thaliacea (n.d.). <https://www.chaloklum-diving.com/marine-life-guide-koh-phangan/fish-reptiles-squirts-chordata/sea-squirts-salps-urochordata/salps-thaliacea/>. Accessed: 2023-06-13.
- Salvesen, Eivind, Aya Saad and Annette Stahl (2020). 'Robust methods of unsupervised clustering to discover new planktonic species in-situ'. In: *Global Oceans 2020: Singapore–US Gulf Coast*. IEEE, pp. 1–9.
- Salvesen, Eivind, Aya Saad and Annette Stahl (2022). 'Robust deep unsupervised learning framework to discover unseen plankton species'. In: *Fourteenth International Conference on Machine Vision (ICMV 2021)*. Vol. 12084. SPIE, pp. 241–250.
- Shannon, Claude Elwood (2001). 'A mathematical theory of communication'. In: *ACM SIGMOBILE mobile computing and communications review* 5.1, pp. 3–55.
- Shorten, Connor and Taghi M Khoshgoftaar (2019). 'A survey on image data augmentation for deep learning'. In: *Journal of big data* 6.1, pp. 1–48.
- Simonyan, Karen and Andrew Zisserman (2014). 'Very deep convolutional networks for large-scale image recognition'. In: *arXiv preprint arXiv:1409.1556*.
- Sosik, Heidi M and Robert J Olson (2007). 'Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry'. In: *Limnology and Oceanography: Methods* 5.6, pp. 204–216.
- Sosik, Heidi M, Emily E Peacock and Emily F Brownlee (2015). 'Annotated plankton images data set for developing and evaluating classification methods'. In: *URL http://darchive.mblwhoilibrary.org/handle/1912/7341*.
- Svetlichny, Leonid, Poul S Larsen and Thomas Kiørboe (2020). 'Kinematic and dynamic scaling of copepod swimming'. In: *Fluids* 5.2, p. 68.
- Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich (2015).

- ‘Going deeper with convolutions’. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.
- Szeliski, Richard (2022). *Computer vision: algorithms and applications*. Springer Nature.
- The Little Plankton Recorder That Could* (n.d.). <https://www.sciencefriday.com/segments/the-little-plankton-recorder-that-could/>. Accessed: 2023-06-13.
- Vinh, Nguyen Xuan, Julien Epps and James Bailey (2009). ‘Information theoretic measures for clusterings comparison: is a correction for chance necessary?’ In: *Proceedings of the 26th annual international conference on machine learning*, pp. 1073–1080.
- Xie, Junyuan, Ross Girshick and Ali Farhadi (2016). ‘Unsupervised deep embedding for clustering analysis’. In: *International conference on machine learning*. PMLR, pp. 478–487.
- Zingone, Adriana, Domenico D’Alelio, Maria Grazia Mazzocchi, Marina Montessor, Diana Sarno et al. (2019). ‘Time series and beyond: multifaceted plankton research at a marine Mediterranean LTER site’. In: *Nature Conservation* 34, pp. 273–310.
- Zooplankton (n.d.). <https://en.wikipedia.org/wiki/Zooplankton>. Accessed: 2023-06-13.

Appendix A

Mathematical Expressions and Derivations

A.1 Approximating the Marginal Distribution

In the context of unsupervised learning and probabilistic modeling, marginal entropy is an important measure that helps in understanding the uncertainty associated with the distribution of a set of labels. As described in Hu et al., 2017, the marginal entropy is calculated using the entire dataset. However, to adhere to memory constraints and computational limitations, the marginalization must be done per batch. Following Dosovitskiy et al., 2014, the marginal distributions is approximated by

$$p_{\theta}(z) \approx \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} p_{\theta}(z | x).$$

Here, $p_{\theta}(z)$ represents the marginal probability distribution of the labels z , and $p_{\theta}(z | x)$ denotes the conditional probability of z given x . In this context, the conditional probability distribution is modeled as an ANN parameterized by θ . The dataset is divided into batches, represented by \mathcal{B} . This approximation is computationally more efficient and is based on the intuition that by averaging conditional probabilities over different subsets (batches) of the data, one can get a good estimate of the overall uncertainty (marginal entropy) in the label distribution.

A.2 Entropy

Marginal and Conditional Entropy

In the study of information theory, entropy is a measure of uncertainty associated with random variables. When dealing with two random variables, X and Z , it is

often necessary to understand the uncertainty of Z given X , and also the uncertainty in the distribution of Y itself. As per Hu et al., 2017, the conditional entropy of random variable Y given X is defined as follows

$$H(Y | X) \equiv \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} H(p_{\theta}(z | x)).$$

Here, $H(Z | X)$ represents the conditional entropy of Z given X . The term $p_{\theta}(z | x)$ represents the conditional probability of z given x , modeled as an ANN with parameters θ . The function H is the entropy function. Following Hu et al., 2017, the marginal entropy of z can be calculated as

$$H(Z) \equiv H(p_{\theta}(z)) = H\left(\frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} p_{\theta}(z | x)\right).$$

Here, $H(Z)$ represents the marginal entropy of Z . The term inside the entropy function is an approximation of the marginal probability distribution of Z , as discussed in Appendix A.1.

A.3 Mutual Information

The derivations presented in this section are cited from Cover, 1999.

Mutual Information and the Entropy

Mutual Information is a measure that quantifies the amount of information obtained about one random variable through observing another random variable. Considering the discrete random variables X and Z . The mutual information between X and Z , denoted as $I(X; Z)$, can be defined in terms of probability mass functions (PMFs) and entropy as presented in Equation (2.4). This expression can be expanded and simplified as follows:

$$\begin{aligned}
I(X; Z) &= \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} p_{(X,Z)}(x, z) \log \frac{p_{(X,Z)}(x, z)}{p_X(x)p_Z(z)} \\
&= \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} p_{(X,Z)}(x, z) \log \frac{p_{(X,Z)}(x, z)}{p_X(x)} - \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} p_{(X,Z)}(x, z) \log p_Z(z) \\
&= \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} p_X(x)p_{z|X=x}(z) \log p_{z|X=x}(z) - \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} p_{(X,Z)}(x, z) \log p_Z(z) \\
&= \sum_{x \in \mathcal{X}} p_X(x) \left(\sum_{z \in \mathcal{Z}} p_{z|X=x}(z) \log p_{z|X=x}(z) \right) - \sum_{z \in \mathcal{Z}} \left(\sum_{x \in \mathcal{X}} p_{(X,Z)}(x, z) \right) \log p_Z(z) \\
&= - \sum_{x \in \mathcal{X}} p_X(x) H(Z | X = x) - \sum_{z \in \mathcal{Z}} p_Z(z) \log p_Z(z) \\
&= -H(Z | X) + H(Z) \\
&= H(Z) - H(Z | X).
\end{aligned}$$

Here, $H(Z)$ is the marginal entropy of Z , representing the uncertainty in Z . $H(Z | X)$ is the conditional entropy of Z given X , representing the uncertainty in Z when X is known.

A.4 Loss Functions

This section provides additional formulas and derivations for the applied methods: IMSAT and IIC.

A.4.1 Information Maximizing Self-Augmented Training (IMSAT)

As presented in Hu et al., 2017, Self-Augmented Training by Virtual Adversarial Training (SAT-VAT) is designed to maximize the robustness of the model's predictions by employing adversarial training. The SAT loss is computed using the following formulation:

$$\mathcal{R}_{\text{SAT}}(\theta; x, T(x)) = - \sum_{m=1}^M \sum_{y_m=0}^{V_m-1} p_{\hat{\theta}}(y_m | x) \log p_{\theta}(y_m | T(x)) \quad (\text{A.1a})$$

$$T(x) = x + r \quad (\text{A.1b})$$

$$r = \arg \max_{r'} \{ \mathcal{R}_{\text{SAT}}(\hat{\theta}; x, x + r'); \|r'\|_2 \leq \epsilon \}. \quad (\text{A.1c})$$

In the above expression, the outer summation iterates over M different classes, and the inner sum evaluates all possible values of a random variable. Here, $T(x)$

is the adversarial perturbation of input x , with r representing the optimal perturbation that maximizes the SAT loss. The quantity ϵ bounds the L_2 norm of the perturbation, ensuring that it is small and not perceptually significant.

When applying SAT-VAT to image classification, certain assumptions enable us to streamline the original formulation. Specifically, for a classification task, we can make the assumption that each instance x_n belongs exclusively to one class. This allows us to reduce the double sum to a single sum over the N instances within the training dataset as follows:

$$\mathcal{R}_{\text{SAT}}(\theta; T) = \frac{1}{N} \sum_{n=1}^N \mathcal{R}_{\text{SAT}}(\theta; x_n, T(x_n))$$

This adapted expression computes the SAT loss for each instance and takes the average across all instances. The SAT loss for an individual instance is solely dependent on the model's predictions corresponding to the true class of that instance. Averaging the individual losses across instances is a common practice in machine learning, especially when utilizing stochastic gradient descent or its variants, as it provides an approximation of the expected loss across the dataset.

Invariant Information Clustering (IIC)

Invariant Information Clustering (IIC) is a method that aims to learn representations of data that are invariant to augmentations. Following Ji et al., 2019, for each augmentation pair i , consisting of an input data point \mathbf{x}_i and its augmentation, we calculate a joint probability matrix \mathbf{P}_i . This matrix is formulated as the outer product of the function $\Phi(\mathbf{x}_i)$ applied to the data point and its augmentation.

$$\mathbf{P}_i = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}'_i)^\top.$$

Here, Φ is a function that maps the data point to a representation space. The joint probability matrix over a batch of data points is obtained by averaging the joint probability matrices of all the augmentation pairs in the batch. For a batch with \mathcal{B} augmentation pairs, this is given by:

$$\mathbf{P} = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_i)^\top.$$

The marginal distributions are needed to compute mutual information. We can calculate them by summing the rows and columns of the joint probability matrix \mathbf{P} . More specifically, \mathbf{P}_c is obtained by summing over all c (rows), and \mathbf{P}'_c by summing over all c' (columns). The final loss function for IIC aims to maximize the mutual information between the representations of augmented data pairs. It is given by:

$$\sum_{c=1}^C \sum_{c'=1}^C \mathbf{P}_{cc'} \cdot \ln \frac{\mathbf{P}_{cc'}}{\mathbf{P}_c \cdot \mathbf{P}_{c'}}. \quad (\text{A.2})$$

Appendix B

Dataset Distribution

Figure [B.1](#) describes the distribution of the AILARON dataset. The dataset contains 21 classes. Note that the dataset contains non-taxonomic classes. Furthermore, some of the classes contain zero instances. The dataset is imbalanced with a majority of dataset instances belonging to the classes copepods and other `Other-irregular`. Figure [B.2](#) describes the distribution of the NDSB dataset. The dataset contains 212 classes. This dataset also contains non-taxonomic classes.

Appendix C

Hyperparameters

Appendix C summarizes the batch size for the architecture utilized in this study. Table C.2 summarizes the hyperparameters utilized in the study, a majority of these parameters are associated with the IMSAT method.

Model	Batch size
VGG	256
DensNet	128
ResNet	512
Inception	256

Table C.1: The batch sizes for machine learning architectures are primarily determined by the model’s size and the computational unit’s memory resources. The batch size is chosen to be as large as possible, given these constraints.

Symbol	Parameter	Value
l	Learning rate	0.001
λ	weighting of the mutual information term	2
ε	step size for the adversarial perturbation	1
ξ	Finite difference approximation constant	
I_p	The number of iterations for computing the perturbation	1

Table C.2: Hyperparameters applied during training. Commonly reported values were used to minimize application-specific tuning.



 **NTNU**

Norwegian University of
Science and Technology