Optimize Plankton Image Classification

Optimizing machine learning algorithms to better classify plankton.

KATRINA LIE

SUPERVISOR
Dr. Nadia Saad Noori
Dr. Marta Moyano

University of Agder, 2024
Faculty of Engineering and Science
Department of Engineering and Sciences

Master

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	Ja
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	Ja
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	Ja
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	Ja
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	Ja
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	Ja
7.	Vi har i flertall blitt enige om at innsatsen innad i gruppen er merkbart forskjellig og ønsker dermed å vurderes individuelt. Ordinært vurderes alle deltakere i prosjektet samlet.	Nei

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	Ja
Er oppgaven båndlagt (konfidensiell)?	Nei
Er oppgaven unntatt offentlighet?	Nei

Acknowledgements

I would like to thank my supervisors, Dr. Nadia Saad Noori and Dr. Marta Moyano. They have both been incredible helpful and supportive throughout the whole project. Their expertise and passion for this field is really shining through. They really helped making this project a higher quality and made me understand challenging concepts.

Secondly I would like to thank my family and friends for always believing in me, maybe a bit too much at times, But it makes it a lot more motivating to work when your being encouraged by the people you love. They may not fully understand what the work I do is, but they cheer me on and help me wherever they can, and I am very grateful for them.

Finally I would like thank Trude Halvorsen, my neurofeedback therapist (which sounds on theme, but is not AI or ML related). She worked efficiently and helped me get the treatment I needed to get healthy enough to write this thesis. She has been incredibly supportive end encouraging, and my body and brain would not have been healthy enough to complete this without her expertise in her field.

Abstract

To understand the impact of climate change on the marine life; researchers have been monitoring and investigating its effect on the food supply chain, among many other things. Plankton form the base of the marine food network and they are key to the sustainability of marine ecosystem. Early life stages are a major bottleneck for marine fish population, and thus understanding the drivers of larval growth and survival is key research topic in fisheries science and marine ecology. Thus, obtaining reliable estimates of the prey fields available (both in terms of abundance, diversity and size) to the fish larvae in the wild can inform about the starvation potential in the larvae, an important mortality source in this stage. In this research work contribute to the on going effort to map zooplankton population and to classify its types based on images captured by a FlowCam instrument for the past decade(2013-2019) and on ZooScan images taken using underwater vision profiler. State of the art image processing and machine learning algorithms had been applied to solve the plankton classification problem. However, with current advances in machine learning and computation power, the task can be optimized the accuracy and improve the results to provide more insights and information about the health and growth of these populations. Some have very high accuracy, while others very low or are being mis-classified - meaning there is still a need for human expertise to verify classification. Even experts can make mistakes, but by optimizing the state-of-the-art methods that are in use classification can happen faster and more precise, supporting research on the fate of marine fish populations under a changing climate. In this project we were able to get a training accuracy of 98% and F1 score of 88% using optimized MobileNet, which is comparative to the state-of-the-art.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Plankton serve an important role in the ecosystem. A study from 2023, [31], stated in their research that plankton not only is important to serve as food for other organisms, but also play a part in transporting carbon in the ocean. Given their role in marine ecosystems, changes in plankton communities will have large impacts in marine biogeochemistry and food webs, among other things. Plankton can be classified into different groups, based on their trophic mode or size. Regarding trophic mode, they can be divided into three broad groups: phytoplankton(photosynthetic organisms), zooplankton(animals) and bacterioplankton(bacteria) [30]. Regarding sizes, they are organized into pico- (0.2-2µm), nano- (2-20µm), micro- (20-200µm), meso- (200µm-2mm), and macroplankton (>2mm).

This research will work to contribute to the on going effort to fully map plankton population, and to classify its types based on images captured by FlowCam instruments for the past decade or so. State of the art image processing and machine learning algorithms had been applied to solve the plankton classification problem in a master thesis by Jan Conradt [7], among others. However, with current advances in machine learning and computation power, the task can be optimized the accuracy and improve the results to provide more insights and information about the health and growth of these populations. Some have very high accuracy, while others very low or are being mis-classified - meaning there is still a need for human expertise to verify classification. Even experts can make mistakes, but by optimizing the state-of-the-art methods that are in use classification can happen faster and better.

1.1 Research question

In this study there are two research questions that are addressed: (1) "Will application of optimization on a convolution neural network improve plankton classification?" (2) "Can weights learned from models trained on plankton images be used to classify different plankton datasets using transfer learning?"

1.2 Research motivation

As for the motivation for this project, the environment is important, and understanding and monitoring the ecosystem in the ocean will be very beneficial to understand how humans effect our planet. Even though plankton are small, they play a big part in the marine ecosystem. There are still large challenges to monitor plankton populations at large scale, and image-based technologies can help advance this field (Goodwin et al. 2022). Given the large amounts of data obtained from these methods, their development needs to go hand in hand with improvements in automatic classification algorithms, ideally unifying methods for different available solutions (e.g. Image FlowCytoBoot, FlowCam, ZooScan, ZooCam)

(Lombardt et al. 2019). Therefore, the motivation for this project is contributing to the automation of plankton monitoring by advancing image classification step.

1.3 Research contribution

The contribution for the research in this study is proving that using transfer learning from our dataset will provide good results on new datasets that have more classes that all are different plankton species from our dataset and give good results. The optimization also proved to provide better results and give a good model to use in other projects for classification and training. And the optimized model can hopefully contribute to better the classification of plankton and map the population of different plankton species.

1.4 Document organization

This study looks at plankton classification and how to improve results. The document is structured in this manner:

- Introduction
  - This chapter introduces the subject and the reason for the study. Here is the research questions stated and what this project have contributed to the ongoing research in the plankton classification field.
- State-of-the-art
  - The results produced by other studies and their approach is presented in an SLR. All the technical information used in this project is also described in this chapter.
- Methods
  - How the datasets are used and how the algorithms are implemented and optimized using transfer learning is described in this chapter.
- Results
  - Here all the results from the benchmarks are presented and explained, divided into two sections, one for the Organized dataset and one for the ZooScan dataset that was shared with us for this project.
- Discussion
  - All the results are discussed here, as well as comparing the results to the state-of-the-art, also the classes that scored low are discussed in more detail.
- Conclusion
  - In the final chapter the findings in this project is concluded and a suggestion for future work is given.

Chapter 2

State-of-the-art

2.1 Background

This research started with a project proposal by Dr. Marta Moyano: "THRESHOLDS: Disentangling the effects of climate-driven processes on North Sea herring recruitment through physiological thresholds" [23]. Getting a better understanding of the prey field for larvae during fall and winter in the North Sea is one of the goals. From this there is a motivation to optimize classification of plankton, from both FlowCam and ZooScan (which was shared to be used in this project by Kanchana Bandara), which is explained in 2.6 and 2.1.6. Further this chapter will go through a literature review of the-state-of-the-art in plankton classification with machine learning, machine learning and the algorithms used and on the dataset.

2.1.1 Literature review

Some papers mention the history of plankton classification and the best result from them, for example the master thesis by Jan Conradt [7] and an article published by Sosa-Trejo et al.[31]. Both papers come to the conclusion that up until 2018 the-state-of-the-art accuracy is from "Texture and Shape Information Fusion of Convolutional Neural Network for Plankton Image Classification" [8] with 96.6% accuracy. As there are several papers released each year, and the papers mentioned above, among others, have conducted a thorough research on the previous history of plankton classification, the main focus on the SLR(Systematic Literature Review) is newly released papers (from 2020) as this will give a good indication to what is being achieved now and how.

As we can see from the SLR in table 2.1 there are some variations in the results depending on the dataset and the algorithms used. There have been high accuracy results up to over 97%, but there is still room for optimization, both with the datasets and the algorithms. And as there are new papers released each year on this topic, the reason to study it and experiment with data and algorithms is high.
All of the papers used CNN or sCNN(Sparse Convolutional Neural Network). The paper by Kerr et al. [20] used multiple algorithms to test the dataset, but achieved best results with models based on Inception and VGGNet. VGGNet was also used by Conradt et al. [6]. Another paper used random forest (Schmidt et al. [29]). From all the papers found they used different methods, but all used supervised learning techniques. Most of them use dataset that are public and all show good, but varying, results. There is not one dataset or one algorithm that is used as a default.

Author (Year)	Source	Title	Keywords	Data	Method	Result
Greer et al.[12] (2023)	Limnology and Oceanography	In situ imaging across ecosystems to resolve the fine-scale oceanographic drivers of a globally significant planktonic grazer	Doliolids, Data processing, marine ecosystem, machine learning	ISIIS (doliolids)	CNN	Recall: 80%, Precision: 89%
Conradt et al.[6] (2022)	Frontiers in Marine Science	Automated Plankton Classification With a Dynamic Optimization and Adaptation Cycle	machine learning, deep neural networks, plankton community, classification, model adaptation	FlowCam	CNN	Recall: 80-90%, Precision: 60-80%
Plonus et al.[26] (2021)	Limnology and Oceanography: Methods	Automatic plankton image classification—Can capsules and filters help cope with data set shift?	automatic classification, deep learning, data set shift, Capsule Network, VPR, plankton image data sets, North Sea	VPR (zooplankton)	CNN	Accuracy: 94.67%
Kerr et al.[20] (2020)	IEEE Access	Collaborative Deep Learning Models to Handle Class Imbalance in FlowCam Plankton Imagery	Feature extraction, Machine learning, Collaboration, Training, Convolution, Computational modeling, Biological neural networks	FlowCam	CNN	Accuracy: 97.4%, F1: 96.2%
Schmidt et al.[29] (2020)	Scientific Reports	Prey and predator overlap at the edge of a mesoscale eddy: fine-scale, in-situ distributions to inform our understanding of oceanographic processes	Bio-oceanography, Image processing, Machine learning, Marine biology, Optical imaging	ISIIS	sCNN	F1: 92%
Swieca et al.[32] (2020)	Marine Ecology Progress Series	Changing with the tides: fine-scale larval fish prey availability and predation pressure near a tidally modulated river plume	Larval fish, Trophic interactions, River plumes, Finescale, Zooplankton distributions	ISIIS	sCNN	F1: 67%, mAP: 77%
Cui et al.[8] (2018)	IEEE	Texture and Shape Information Fusion of Convolutional Neural Network for Plankton Image Classification	Feature extraction, Shape, Convolutional neural networks, Databases, Image enhancement, Ecosystems	WHOI	CNN	Accuracy: 96.6%

Table 2.1: Summary of recent studies

2.1.2 Approaches

Machine Learning Machine learning is a technique to make a computer learn through experience, [16]. It is given input that can be all types of data and learns in different ways, depending on what method and algorithm is used. Supervised and unsupervised learning are the two most common methods used.

Supervised:
Supervised learning is training data where the input is data that has a label and the goal is to predict that the input has the correct label [16]. The prediction of this method often comes from a formed decision tree.

Unsupervised:
Using unsupervised learning means the data that is analysed is unlabeled. This is often used when there are assumptions about the data structural properties [16].

2.1.3 Image Classification

When using image classification the goal is to put an label to an image that makes sense. Figure 2.1 shows typical stages of image classification with convolutional neural network (CNN), an image is used as the input, then there are x-amount of convolutional and pooling layers, the output from this is then feed into fully connected layers and the final layer gives the label of the image as an output [27].

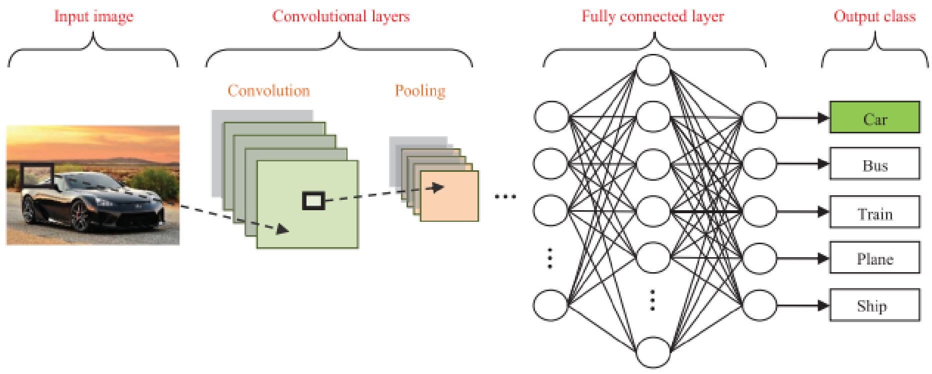


Figure 2.1: Image Classification architecture [27].

Convolution:
The convolution layers are used to extract the features of the input and compress the input information [27]. Feature maps are then created by a group of neurons, each of these neurons are connected to a group of neurons in the layer prior to the one they are in. Weights are added between these connections, then with learned weights new feature maps are created. Weights for neurons in the same feature map are equal, but feature maps in the same layer can have different weights.

Pooling:
Pooling is used to reduce the resolution of feature maps [27]. It can be done by maximum, minimum or average pooling. Pooling is explained in figure 2.2 where you can see with the max pooling example the highest feature value in the region (that is chosen by the filter size) is selected, and it creates a reduced feature map that takes less computational power and

time to process.

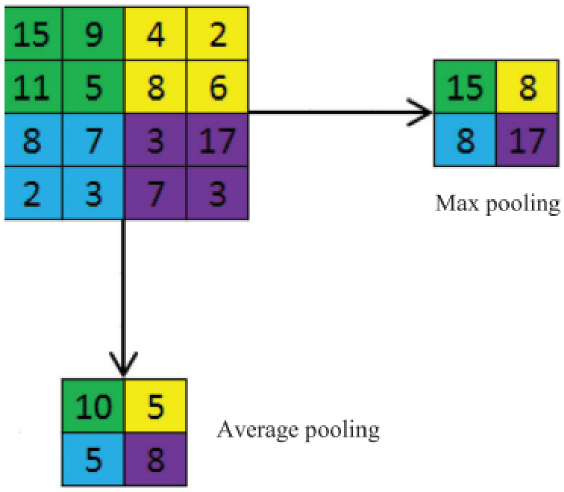


Figure 2.2: Pooling example [27].

Fully Connected Layers:
The fully connected layers function is to interpret the features from the convolution and pooling layers and use a form of reasoning on them [27]. An activation function is often used on the final fully connected layer to do the final classification or labeling.

Activation function:
In neural networks the activation function defines the output of a node [22]. ReLu (rectified linear unit) is one activation function that takes the input from the model, if the input has negative value the output is zero, if the input has positive value the output is the same as the input. Softmax is another function that works a bit different from ReLu. It works with taking the output vector and transform the values to probability values that can be used to predict the output neurons[4]. This is often used for multiclass classification in the final layer.

Optimizer:
After loss is calculated the weights will be updated during backpropagation. An optimizer is used in backpropagation to try and converge the loss function to a minimum [9]. Adam (Adaptive Moment Estimation) is an optimizer that uses adaptive learning rates, has bias correction and does computation based on first and second gradient moments [21] which makes it more efficient.

Transfer learning:
Transfer learning is mostly used to train networks where there is not enough data to train from scratch [5]. This method loads pre-trained weights that have learned from a different model. There is an option to freeze the base model during training to avoid altering information during training, and only update the weights on the new trainable layers that are added after the base model is loaded.

2.1.4 Algorithms

The algorithms described in this sections are the ones that will be focused on in this project. There are several ML algorithms that works very well with object detection, but these were the ones that seemed interesting to try or to optimize for plankton classification. DenseNet and MobileNet also showed promising results in the bachelor thesis that curated the dataset and benchmarked it [35]. All of the algorithms mentioned below are supervised.

DenseNet:
DenseNet is a convolutional neural network (CNN) where each layer receives input from all of the previous layers, in contract to standard CNN where each layer get input from the previous layer [10]. Skip connections, which allows propagation of information between layers, is used to mitigate a vanishing gradient problem. DenseNet is built by creating DenseBlocks and it is in the blocks that the convolution is applied, among with other techniques chosen for the block. Between the blocks are Transition layers that performs convolution and pooling to reduce the feature maps for the next block [28].

MobileNet:
MobileNet is based on factorised convolutions, called depthwise separable filters [13]. In contrast to regular convolution, depthwise convolution separates the filter that is added to the input and the combining instead of doing it one step, reducing both size and computational cost. Each of the layers are followed by BN (batch normalisation) and ReLu (rectified linear activation function), the final layer do not have this, but is feed into a softmax layer for classification. How the layers are built can be seen in figure 2.3.

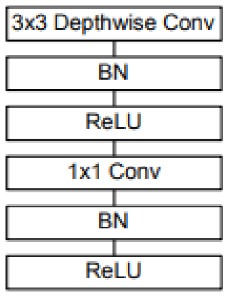


Figure 2.3: Layers in MobileNet [13].

Siamese network:
Siamese network uses a technique called similarity problems, where the outputs goal is to find if two inputs are similar or not [24]. If the two inputs are similar the output will be a high score, and low if they are not similar. A Siamese network can build a good model from only one picture of the class to classify it, called one shot learning. Figure 2.4 shows the architecture of a Siamese network that consist of more than one network that are identical (use same weights and parameters). To compare images the input is sent to one of the networks to get a vector representation of the image, the image to compare with is sent through another network and then distance between the vectors can be measured. If the distance is small then the inputs are similar, and if the distance large then they are not similar. During training the network uses a loss function for evaluating if it is seeing if two images are similar or not.

2.1.5 Metrics

To be able to evaluate the algorithms learning and how well it is performing it is important to use some metrics that tells us whether it is predicting as intended or if there are false predictions.

F1 Score:

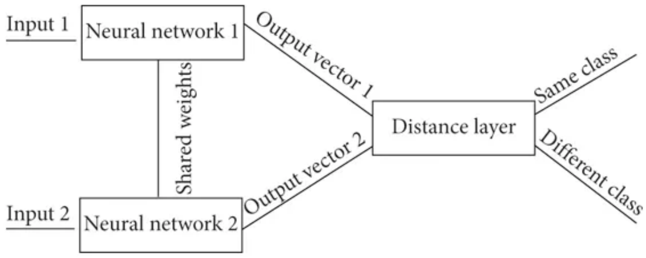


Figure 2.4: Siamese architecture [24].

The F1 score takes the harmonic mean of precision and recall [14], as seen in figure 2.5. Precision a measure of true positives from all positive cases (True Positive/(True Positive+True Negative)). Recall is a measure of true positive from the actual positive cases (True Positive/(True Positive+False Negative)). This is a metric that gives a good indication of how the model is predicting, and can be very helpful if there is class imbalance. In this paper F1 macro and F1 micro will be used. The macro score calculates the F1 score for every class and gives an F1 average, the micro score is a global average which takes all True Positives, False Negatives and False Positives and calculates the F1 score.

$$\text{F1-score} = \left(\frac{\text{Recall}^{-1} + \text{Precision}^{-1}}{2}\right)^{-1} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

Figure 2.5: F1 Score calculation [14].

Accuracy:
Accuracy is calculated by taking all the correctly predicted cases and dividing them by the total number of cases, [14]. This gives and overall picture of the correctness of the model, especially if there is no class imbalance in the data, and it does not take account of false negatives.

Loss:
During training a loss function is common to add as it can be used to evaluate and fit the model[36]. The goal is to have zero, or as little loss as possible. During training the hyperparameters (weights and biases) are used in a loss function (*Average Loss* is explained in the article [36]) and after the loss is calculated the hyperpartameters are adjusted to get the loss value lower.

2.1.6 Data

Most relevant for this project is images of plankton that are categorised and can be used to train algorithms. FlowCam was the technology used for getting the images, and Dr. Marta Moyano is providing this project the images. However there are other technologies used to get plankton images, ZooScan being one of them. Images provided by ZooScan can become available for this project and can be used for further testing the classifiers.

FlowCam

Plankton have a size that is microscopic they require special equipment to be captured in images. The images used for this project are, as previously mentioned, captured using FlowCam. FlowCam uses flow imaging microscopy, [33], which allows the camera to capture microorganisms and particles flowing in a microfluidic channel. Figure 2.6 displays the components used, the flow cell is where the particles and microorganisms are placed using a syringe, [34]. To help the camera capture the organisms that are moving it use strobe LED to illuminate the source, combined with a camera that has a very short shutter speed. Currently FlowCam can count and characterise particles from 300nm to 5mm, using different lenses for different sizes.

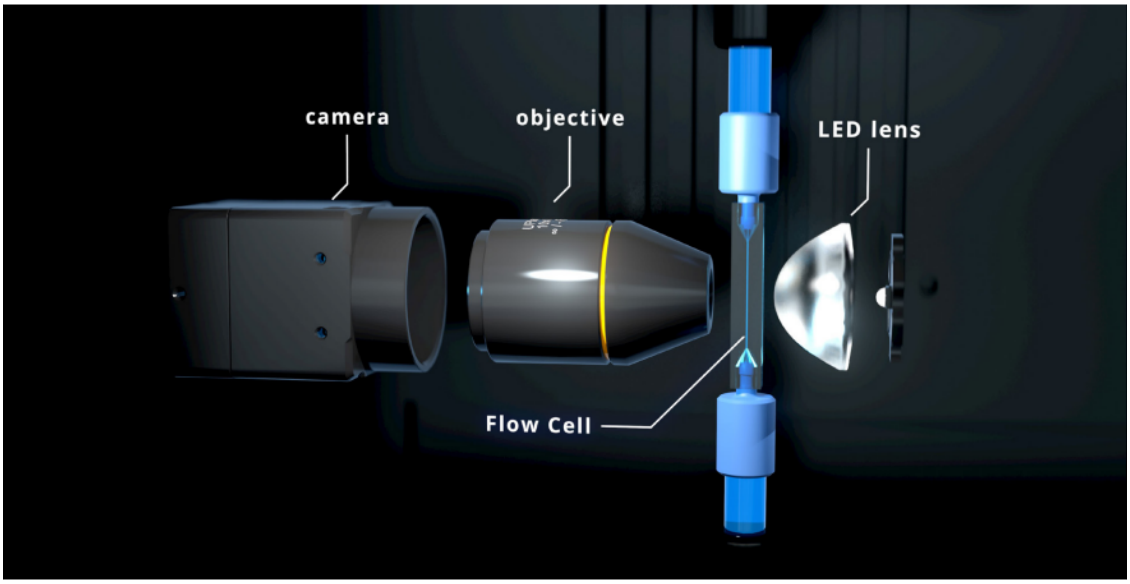


Figure 2.6: The different components used in flow imaging microscopy, [34].

Zooscan

ZooScan is built to be able to safely process liquid samples, consisting of a draining device and a high resolution imaging device and has a resolution of $10.6\mu m$[11]. It is suited for organisms that are larger than $200\mu m$ and should not scan over 1500 objects in one frame. After a scan is completed the image is normalized, converted to grey scale, removing background and edges, then the individual objects are extracted and measured. How it is set up and the frame used for the liquid and imaging is seen in figure 2.7.

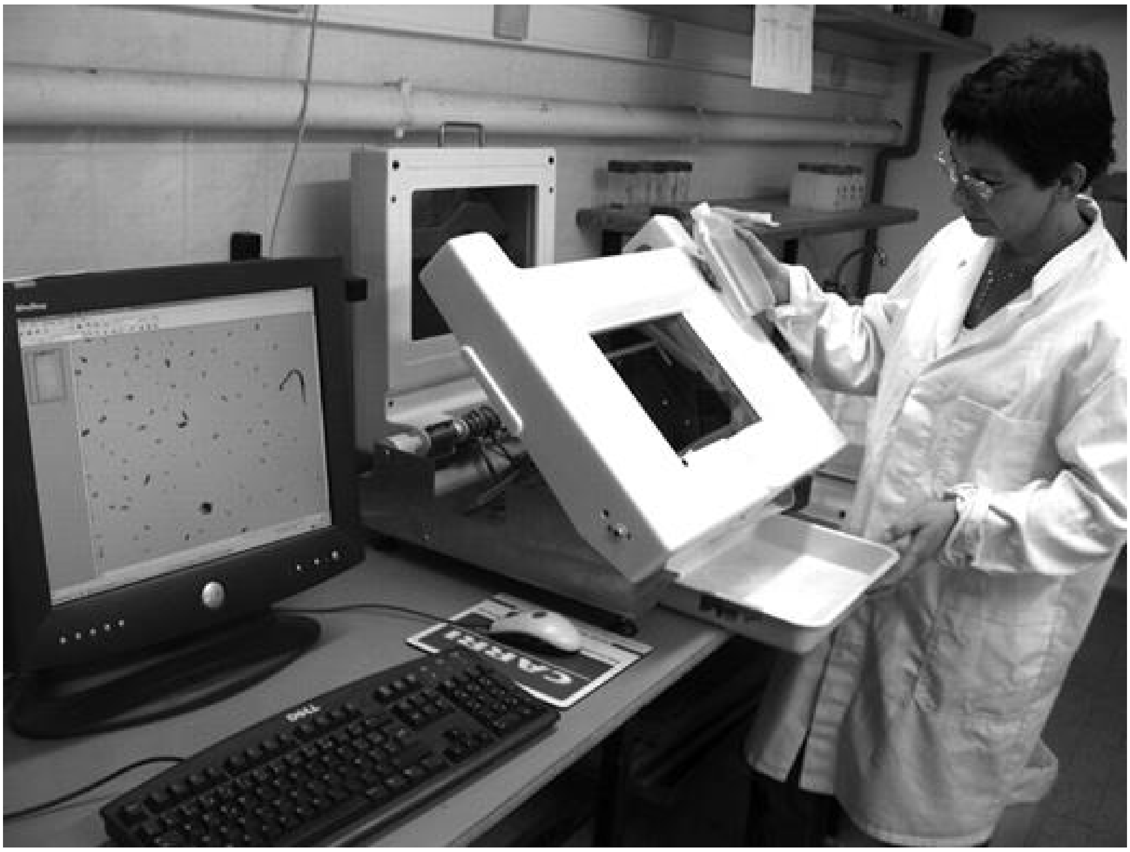


Figure 2.7: ZooScan setup example, [11].

Organized Dataset

The dataset was cleaned and tested by a previous study, [35]. They got the images from Dr. Marta Moyano and there were several hundred thousands images. After cleaning the dataset, removing unusable images and duplicates, the bachelor group was left with 12 categories and around 372 246 images. In their final dataset they augmented the images with flipping and rotating them, which made the final number of images in the dataset 505 421. Figure 2.9 shows the distribution of plankton images in the different categories, and figure 2.8 shows one image from each class.

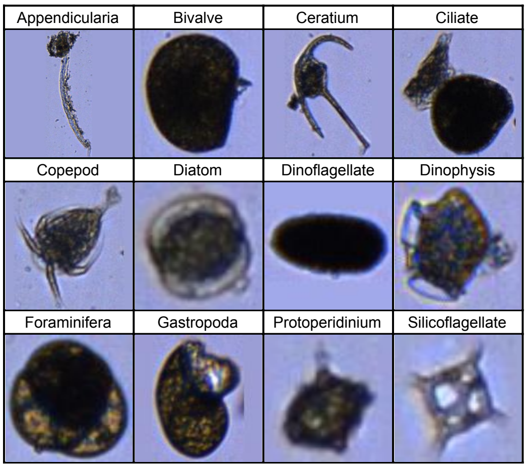


Figure 2.8: Example of each plankton classes.

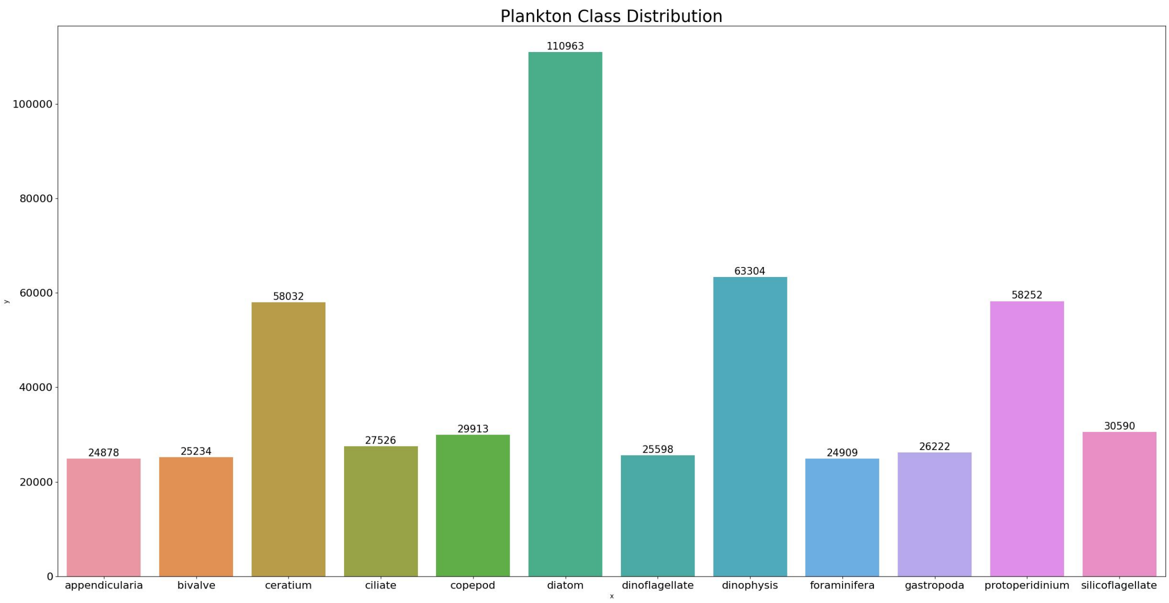


Figure 2.9: Image distribution of plankton classes.

Kanchana - ZooScan dataset

Kanchana Bandara shared a dataset consisting of 73 classes of plankton. The dataset is based on plankton samples that was published by "SAENOE" *(Sea Scientific Open Data Publication)*, [1]. ZooScan was used to get the images, and the full dataset consists of 93 taxa. Further the dataset was modified by adding images Kanchana Bandara and his team had available for their project to certain classes, and adding and removing some classes based on how relevant they were for the project. Now the dataset shared to be used in this project has 73 classes, some of them being non-living categories. From this dataset the team only used 62 classes (the ones they found most relevant) and had an accuracy and F1 score of more than 91%, [2].

In this project this dataset will be used to get a baseline for ZooScan images, and see how the models perform and see if the result can be improved by optimizing the algorithms. Figure 2.11 shows the distribution between the classes. There is some class imbalance, with *Detritus* having more than double the data as other classes.

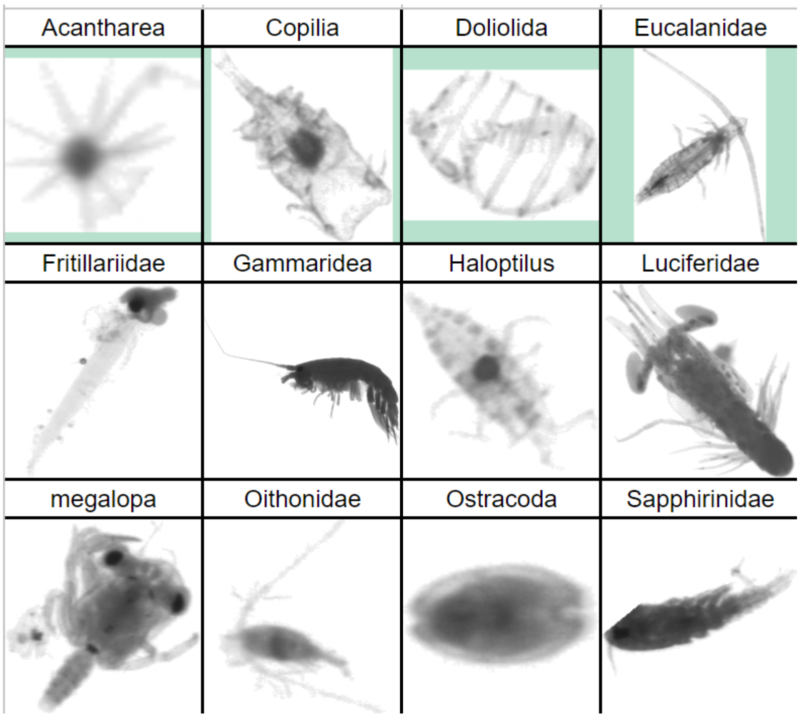


Figure 2.10: Example of some of the living classes.

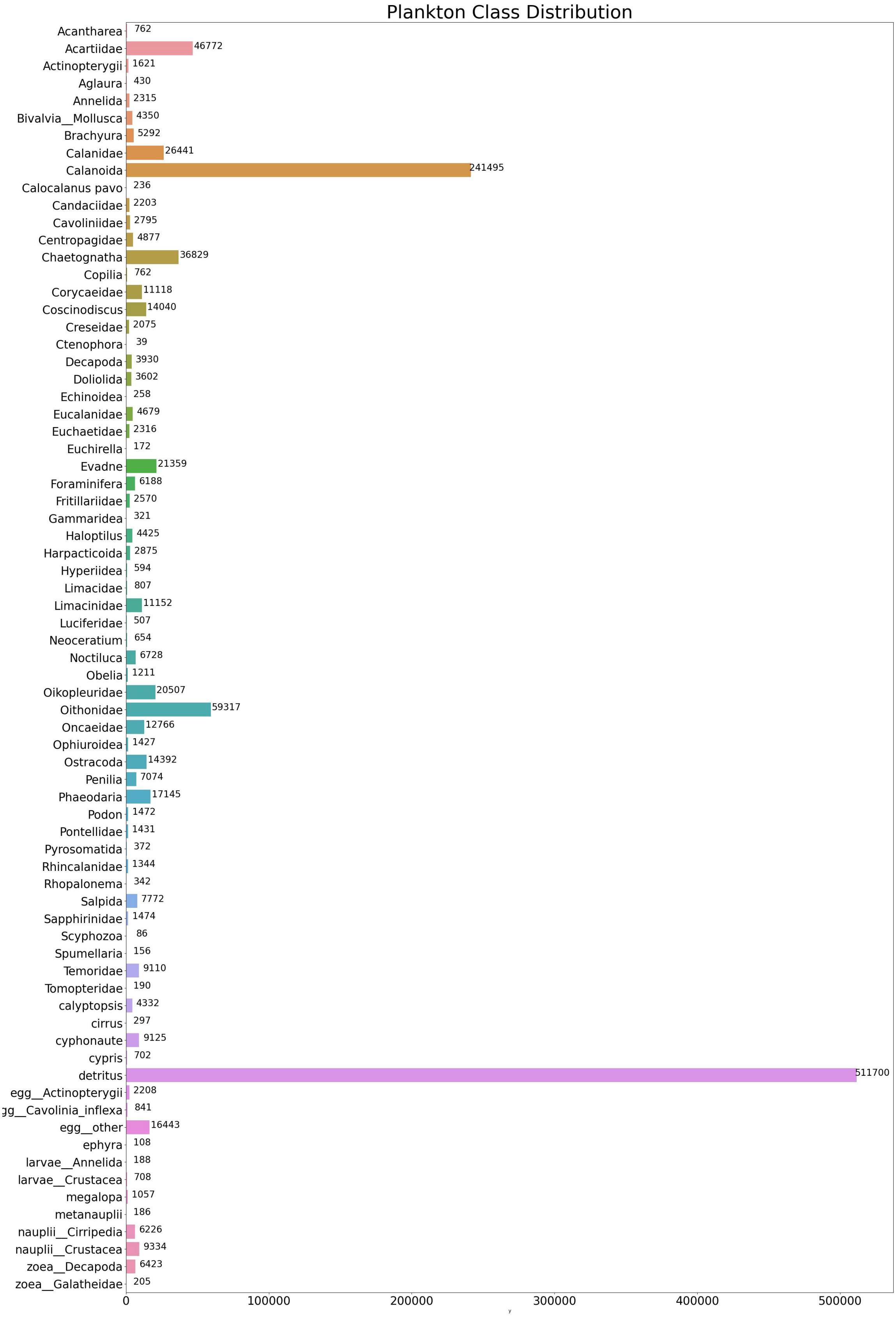


Figure 2.11: Image distribution of plankton classes.

2.1.7 Tools

For this project a lot of the tools described in this section were used to be able to reproduce the results from the previous bachelor group.

Python
Programming language used is python, as it is easy to work with, large community and has a lot of libraries and tools for machine learning. This includes Keras 2.1.7, TensorFlow and Scikit-learn.

JupyterHub
One of the software used for programming was JupyterHub. JupyterHub combines Jupyter Notebook and JupyterLab in a web interface, [17]. Which means training algorithms is done using the computational power in JupyterHub, and is accessible for everyone without any worry of the hardware specification of a personal computer. The JupyterHub used in this project is powered by UiA and has some limitations.

Weights & Biases
Weights & Biases is a tool for developing machine learning models, making it easy to visualise training data and provides reports for documentation [3]. In this project it is used to create tables and charts of the training.

Keras
Keras is an API used for deep learning, [18]. This API offers deep learning models with pre-trained weights, including EfficientNet, MobileNet and DenseNet [19].

Chapter 3

Methods

The methods used in this project have been to optimize the results from two different datasets. Removing images and classes was done for the ZooScan dataset. To optimize the algorithm the proposed method was to add more Fully Connected (FC) layers to make extract more features during training. This was used on the MobileNetLarge and DenseNet169. Some hyperparameters were also changed to test what provided best results. The sections below will go into further detail about the methods. The code used for the project can be accessed on this GitHub repository: Optimize Plankton Classification, here the confusion matrix will be stored as well under "plankton_classification/results to see the image clearer if needed.

3.1 Datasets

3.1.1 Organized_v7

This dataset has already been tested and used in previous projects. Here it will be used to test different algorithms and see if they show better performance than the ones already used (DenseNet, MobileNet, and more in the bachelor thesis "Towards Better Plankton Classification" [35]).

First the results were reproduced to verify that the dataset and algorithms provide validated results, and use this to compare with the results achieved with optimization and different algorithms.

3.1.2 ZooScan

The dataset was split into in train, test and validate with the ratio 0.8/0.1/0.1. Then it was loaded into the code as a TensorFlow dataset object. As there were 73 classes, some of them were removed as they were not that relevant, the "egg_" categories were removed first. However a it is desired to keep as many classes as possible to create a more robust and usable model.

Then a more balanced dataset was created using a maximum of 15k images from each of the 73 original classes using the

ds_balancer_v1.py python code. This was to create a smaller dataset to use for testing as the original one was very unbalanced. Some classes with lower number of samples would keep their original total.

3.2 Algorithms

3.2.1 DenseNet169

This network uses transfer learning, the model used was DenseNet169 which is imported from *tensorflow.keras.applications*, and the weights where from *imagenet*. After the base model is implemented a global pooling layer and bacthnormalization is added, then dropout layers to prevent overfitting (before and after FC layers) and the final FC layer for prediction uses softmax as the activation function. The optimizer used is Adam and the loss is "categorical_crossentropy".

3.2.2 MobileNetLarge

MobileNet also use transfer learning, with the model and weights imported from same place as DenseNet. The setup for the model after the base network is similar to DenseNet, but without the dropout layer.
To use the weights already learned from the "Organized" dataset, the *best_model.h5* was added to the python script. Then *by_name=True* and *skip_mismatch=True* was added to the */.local/lib/python3.10/site-packages/keras/applications/mobilenet_v3.py* file as this made it possible to use custom weights where the layers may not match completely.

```
    # Load weights.
    if weights == "imagenet":
        ...
         model.load_weights(weights_path)

    elif weights is not None:
         model.load_weights(weights,by_name=True, skip_mismatch=True)
```

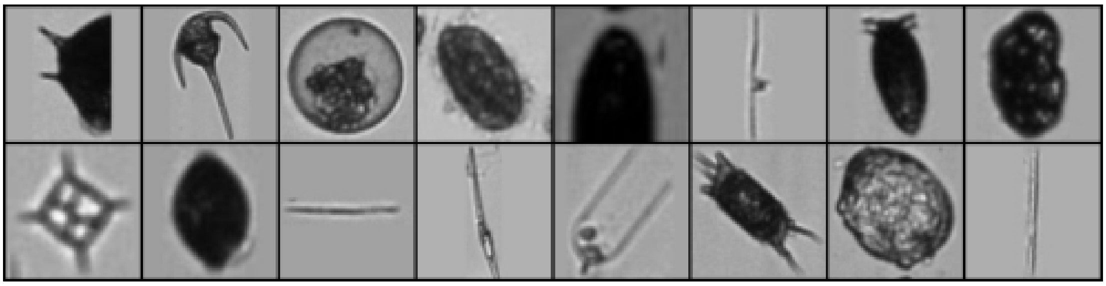
3.2.3 Siamese

Siamese network have the potential to learn on small data and could be interesting to test on these datasets. This was tested by using two different githubs: siamese by Data Hacker, referred to as "014_siameseNetwork.ipynb" in the github under "models/siamese" and MNIST siamese, referred to as "siamese_mnist.ipynb".
For the first one it creates a dataset by creating pairs with a label, this label is to tell whether they are in the same class or not. Figure 3.1 shows an example of pairs and their labels. Then the siamese network is created with CNN layers and FC layers, where both images from the pairs are passed into. This network uses "contrastive loss" and "adam" for optimizer. When it runs through the epochs it calculates the loss and distance between two images.
The second one uses the MNIST dataset to load the images when creating pairs. Then create the base network using "contrastive loss" and "rms" as the optimizer. Finally the pairs are used for input in the network and it uses it to train and creates a model used for classification.

3.2.4 Optimizing

Different datasets needs different optimizing and tuning to achieve better scores. The goal is go get a model that recognize more features and learn more complex combinations. This was mainly done by adding more FC layers and adjusting the batch size.

Fully Connected Layers
The original code had two FC layers. To add more complexity to the network, two more layers were added. First three were added, but that showed little to none improvement. Four



[1. 1. 0. 1. 1. 0. 1. 1.]

Figure 3.1: An example of pairs from the Organized dataset in the siamese network by Data Hacker, label "1" means they are not in the same class, label "0" means they are from the same class.

layers improved the results, which can be seen in the Results chapter 4. Adding more layers was not tested, but could result in overfitting and require too much computational power. The code below shows how the layers were configured using ReLu as the activation function and having a lower input size for each layer.

```
# fully connected layers for intermediate features
    x= Dense(1024,activation='relu')(x)
    x= Dense(512,activation='relu')(x)
    x= Dense(256,activation='relu')(x)
    x= Dense(128,activation='relu')(x)
```

Batch Size

The batch size determines how many samples from the dataset is being used for predictions before calculating the error, [25]. This project uses "Mini Batch Gradient Descent", which often use a size of 32, 64 or 128. Using higher or lower batch-size can make the model converge faster, but also add more oscillation. In this project a batch size of 32 was the standard, but both 64 and 128 was also used to see whether it would provide better results.

Chapter 4

Results

This chapter will look at the results produced from the organized_v7 dataset and the ZooScan dataset. There were multiple benchmarks done for each dataset, each resulting equal or better accuracy or F1 score. The final results come from trial and error on hyper-parameters and information based on previous studies.

4.1 Organized_v7

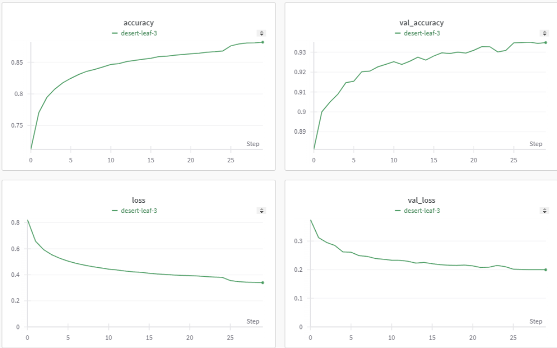
As this dataset have been benchmarked in previous studies, the first benchmark in this report was to verify that the results could be reproduced. Then try to better the results with optimizing.

4.1.1 First Benchmark

Table 4.1 shows the results gathered from training using transfer learning. MobileNet has a better results. In figure 4.1 we see the accuracy and loss, where MobileNet shows signs of overfitting, while DenseNet has a more smooth curve. However, MobileNet converges faster and is more efficient to use for training. The normalized confusion matrix in figure 4.2 shows how well each class was predicted, and have very similar results on both networks. There are some classes that have more mis-classification(false positives and false negatives), *bivalve* and *dinoflagellate* in particular, while *ceratium*, *copepod*, *dinophysis* and *silicoflagelatle* have very promising results.

Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
organized_v7	densenet169	32	12	0,88	0,94	0,94	0,80
organized_v7	mobilenetL	32	12	0,95	0,96	0,96	0,88

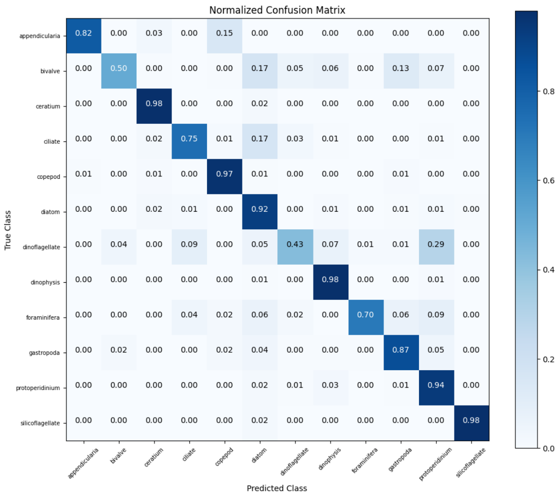
Table 4.1: Results from training the organized dataset on densenet and mobilenet.



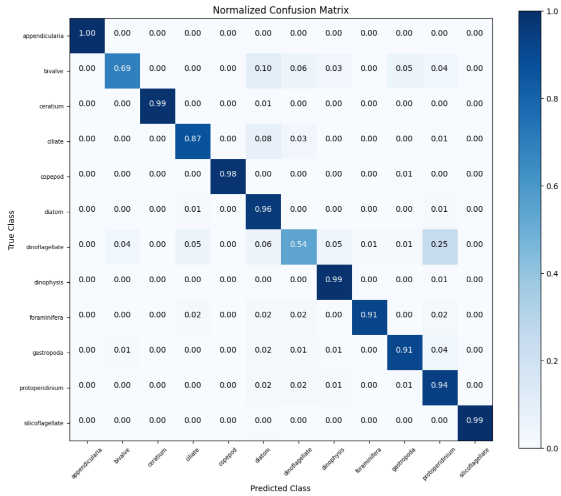
(a) DenseNet169

(b) MobileNetV3Large

Figure 4.1: Metrics during the first benchmark training.



(a) DenseNet169



(b) MobileNetV3Large

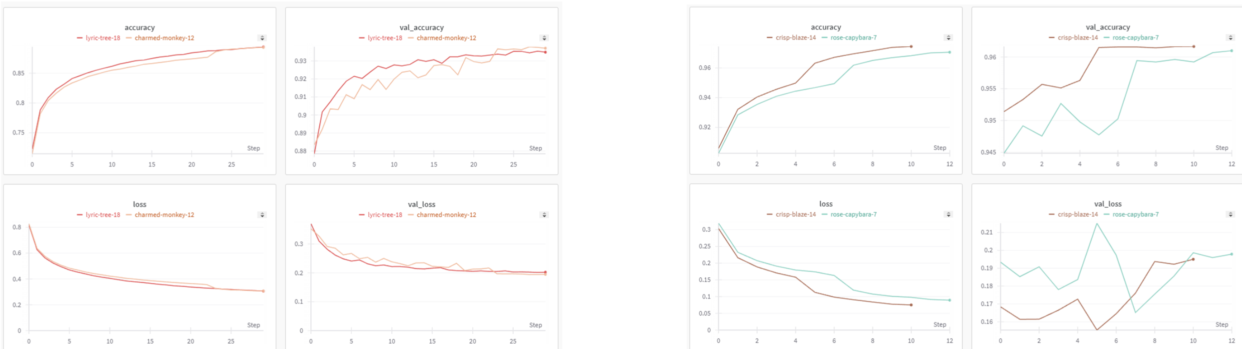
Figure 4.2: Normalized confusion matrix for the first benchmark.

4.1.2 Second Benchmark

In this test both densenet and mobilenet have been optimized by adding two more FC layers, as described in the optimizing section in "Methods" 3.2.4, and used a batch size of both 32 and 64. Table 4.2 shows that Densenet went up one percent in training accuracy and none in validation and testing. Mobilenet went up two-three percent in training accuracy, but also stayed the same in validation and testing. The F1 score on both had + 0.01. Figure 4.3 shows similar graphs to the first benchmark, but mobilenet has a bit more signs of overfitting on the validation graph. The confusion matrix, in figure 4.4 is also similar as the last test, with some mis-classifications going down, while others went a bit up.

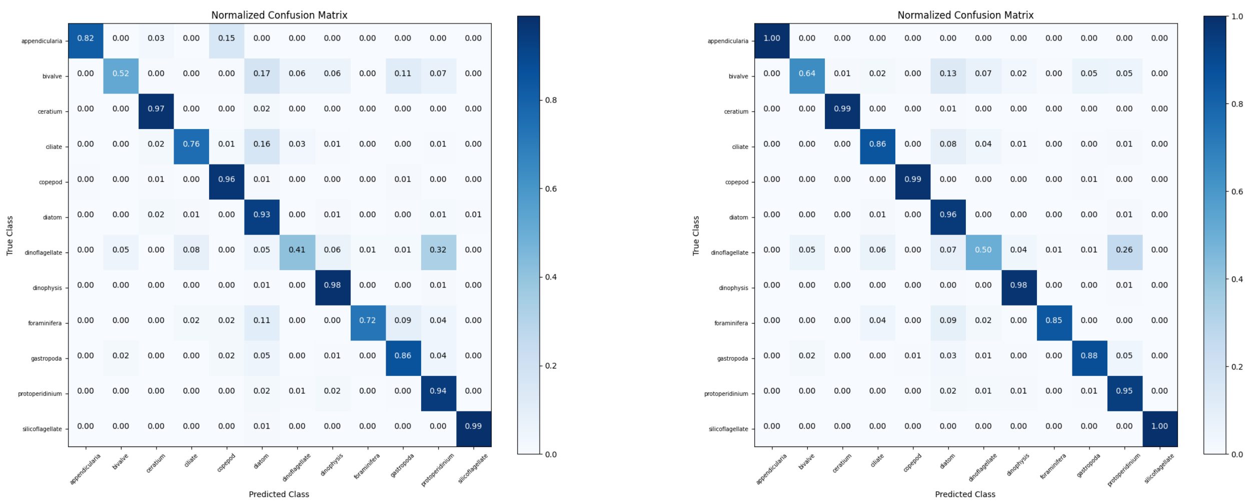
Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
organized_v7	densenet169Opt	32	12	0,89	0,94	0,94	0,81
organized_v7	densenet169Opt	64	12	0,89	0,94	0,94	0,79
organized_v7	mobilenetLOpt	32	12	0,97	0,96	0,96	0,87
organized_v7	mobilenetLOpt	64	12	0,98	0,96	0,96	0,88

Table 4.2: Results from training the organized dataset on optimized densenet and mobilenet(four FC layers (1024,512,256,128).



(a) DenseNet169 (b) MobileNetV3Large

Figure 4.3: Metrics during second benchmark training.



(a) DenseNet169 (b) MobileNetV3Large

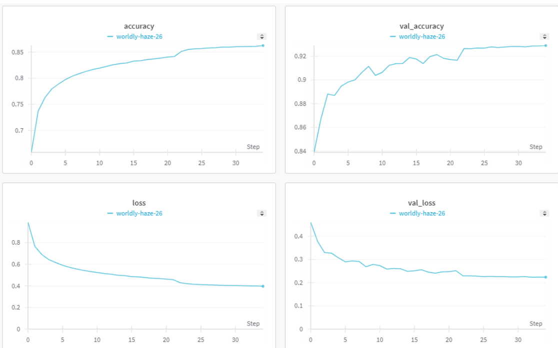
Figure 4.4: Normalized confusion matrix for the best results in the second benchmark.

4.1.3 Third Benchmark

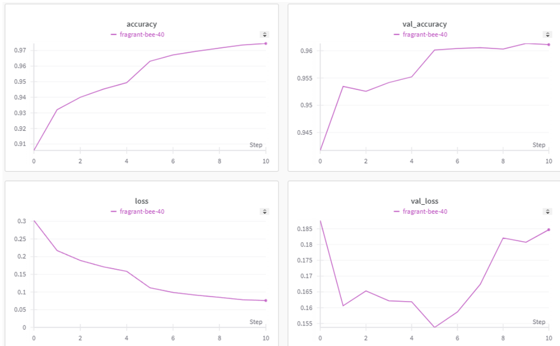
For a final benchmark on this dataset the trained weights from the best model created from the ZooScan dataset (second benchmark, MobileNet, 128 batch size, 4.6) was used instead of "imagenet" pre-trained weights. DenseNet does worse than in the second benchmark, with an F1 score of 77%, which is the lowest on all the tests on this dataset. MobileNet had similar results as in the previous benchmark, showing signs of overfitting during validation in figure 4.5, and a validation loss that does not converge. The confusion matrix in figure 4.6 also shows the same classes as the previous benchmark are being mis-classified. Comparing MobileNets graphs with the previous benchmarks it has less overfitting with a higher batch size.

Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
organized_v7	densenet169Opt	64	12	0,86	0,93	0,93	0,77
organized_v7	mobilenetLOpt	64	12	0,97	0,96	0,96	0,88

Table 4.3: Results from training the organized dataset on optimized DenseNet and MobileNet.

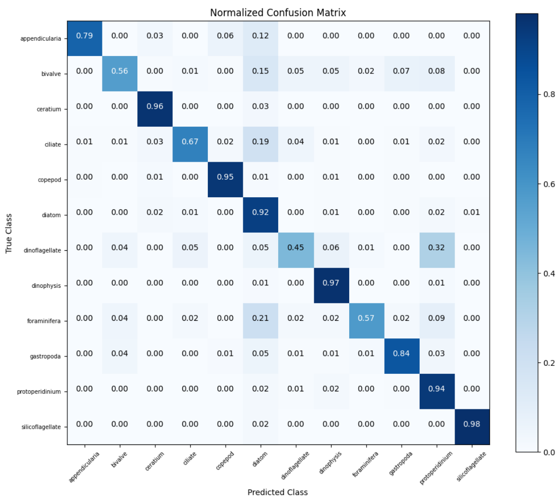


(a) DenseNet169

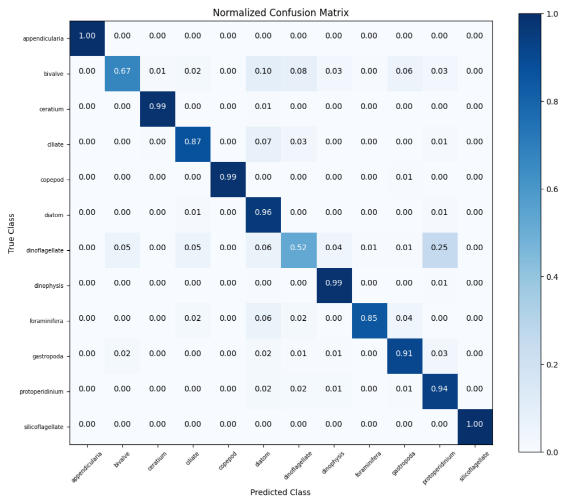


(b) MobileNetV3Large

Figure 4.5: Metrics during third benchmark training.



(a) DenseNet169



(b) MobileNetV3Large

Figure 4.6: Normalized confusion matrix for the results in the third benchmark.

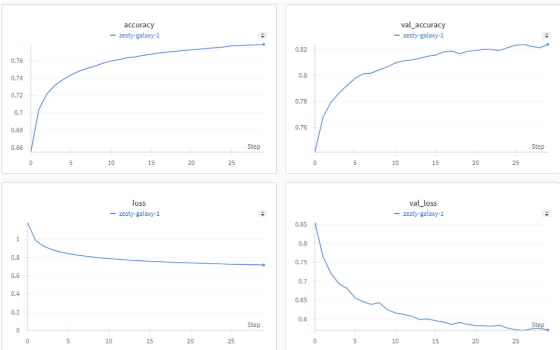
4.2 ZooScan

4.2.1 First Benchmark

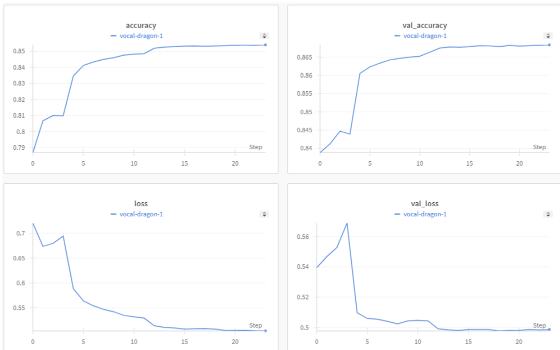
As a first test in the ZooScan dataset nothing was altered in the dataset, all 73 classes were used, and the original setup of DenseNet and MobileNet was used. The results in table 4.4 show that the accuracy and F1 score is on the lower side, with MobileNet again giving the best results. The curves in figure 4.7 shows that DenseNet had a smooth line, however as it ran for all 30 epochs it was set at, it might would benefit from a higher number of epochs. MobileNet shows some signs of overfitting, but converges fast. The confusion matrix is included in figure 4.9 as it shows that some classes are highly mis-classified, those being mis-classified as the two classes that have a high imagecount as seen in figure 2.11, *calanoida* and *detritus*.

Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
ZooScan	densenet169	32	73	0,78	0,82	0,82	0,49
ZooScan	mobilenetL	32	73	0,85	0,87	0,87	0,62

Table 4.4: Results from training the ZooScan dataset on DenseNet and MobileNet.

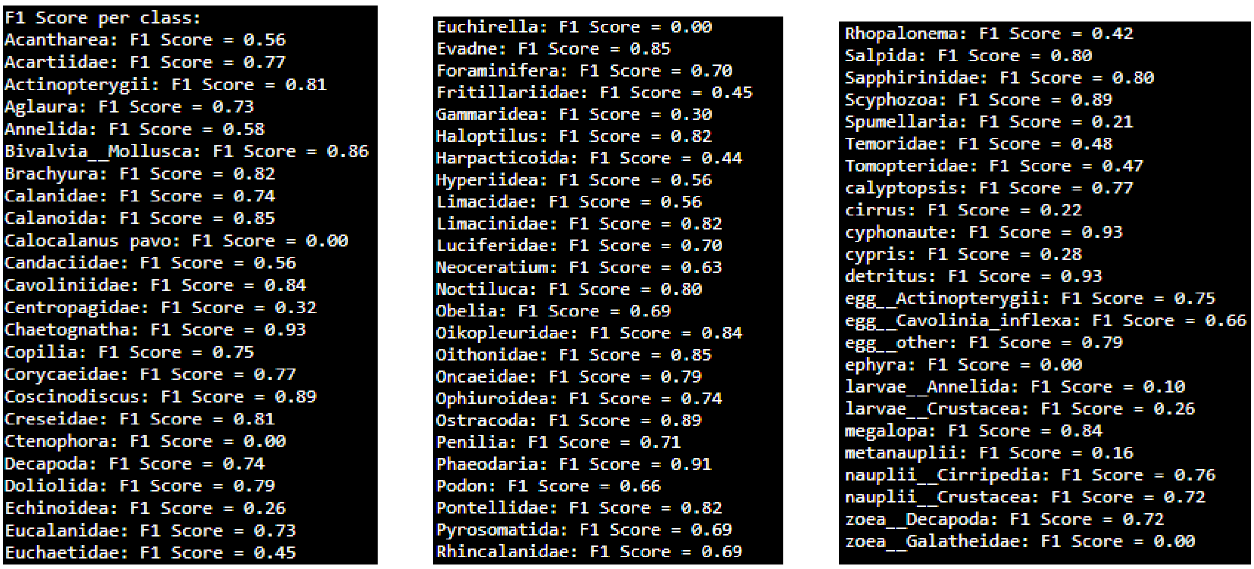


(a) DenseNet169



(b) MobileNetV3Large

Figure 4.7: Metrics during the first benchmark training.



```
F1 Score per class:
Acantharea: F1 Score = 0.56
Acartiidae: F1 Score = 0.77
Actinopterygii: F1 Score = 0.81
Aglaura: F1 Score = 0.73
Annelida: F1 Score = 0.58
Bivalvia__Mollusca: F1 Score = 0.86
Brachyura: F1 Score = 0.82
Calanidae: F1 Score = 0.74
Calanoida: F1 Score = 0.85
Calocalanus pavo: F1 Score = 0.00
Candaciidae: F1 Score = 0.56
Cavoliniidae: F1 Score = 0.84
Centropagidae: F1 Score = 0.32
Chaetognatha: F1 Score = 0.93
Copilia: F1 Score = 0.75
Corycaeidae: F1 Score = 0.77
Coscinodiscus: F1 Score = 0.89
Creseidae: F1 Score = 0.81
Ctenophora: F1 Score = 0.00
Decapoda: F1 Score = 0.74
Doliolida: F1 Score = 0.79
Echinoidea: F1 Score = 0.26
Eucalanidae: F1 Score = 0.73
Euchaetidae: F1 Score = 0.45
```

```
Euchirella: F1 Score = 0.00
Evadne: F1 Score = 0.85
Foraminifera: F1 Score = 0.70
Fritillariidae: F1 Score = 0.45
Gammaridea: F1 Score = 0.30
Haloptilus: F1 Score = 0.82
Harpacticoida: F1 Score = 0.44
Hyperiidea: F1 Score = 0.56
Limacidae: F1 Score = 0.56
Limacinidae: F1 Score = 0.82
Luciferidae: F1 Score = 0.70
Neoceratium: F1 Score = 0.63
Noctiluca: F1 Score = 0.80
Obelia: F1 Score = 0.69
Oikopleuridae: F1 Score = 0.84
Oithonidae: F1 Score = 0.85
Oncaeidae: F1 Score = 0.79
Ophiuroidea: F1 Score = 0.74
Ostracoda: F1 Score = 0.89
Penilia: F1 Score = 0.71
Phaeodaria: F1 Score = 0.91
Podon: F1 Score = 0.66
Pontellidae: F1 Score = 0.82
Pyrosomatida: F1 Score = 0.69
Rhincalanidae: F1 Score = 0.69
```

```
Rhopalonema: F1 Score = 0.42
Salpida: F1 Score = 0.80
Sapphirinidae: F1 Score = 0.80
Scyphozoa: F1 Score = 0.89
Spumellaria: F1 Score = 0.21
Temoridae: F1 Score = 0.48
Tomopteridae: F1 Score = 0.47
calyptopsis: F1 Score = 0.77
cirrus: F1 Score = 0.22
cyphonaute: F1 Score = 0.93
cypris: F1 Score = 0.28
detritus: F1 Score = 0.93
egg__Actinopterygii: F1 Score = 0.75
egg__Cavolinia_inflexa: F1 Score = 0.66
egg__other: F1 Score = 0.79
ephyra: F1 Score = 0.00
larvae__Annelida: F1 Score = 0.10
larvae__Crustacea: F1 Score = 0.26
megalopa: F1 Score = 0.84
metanauplii: F1 Score = 0.16
nauplii__Cirripedia: F1 Score = 0.76
nauplii__Crustacea: F1 Score = 0.72
zoea__Decapoda: F1 Score = 0.72
zoea__Galatheidae: F1 Score = 0.00
```

Figure 4.8: The F1 score of each class from testing MobileNet.

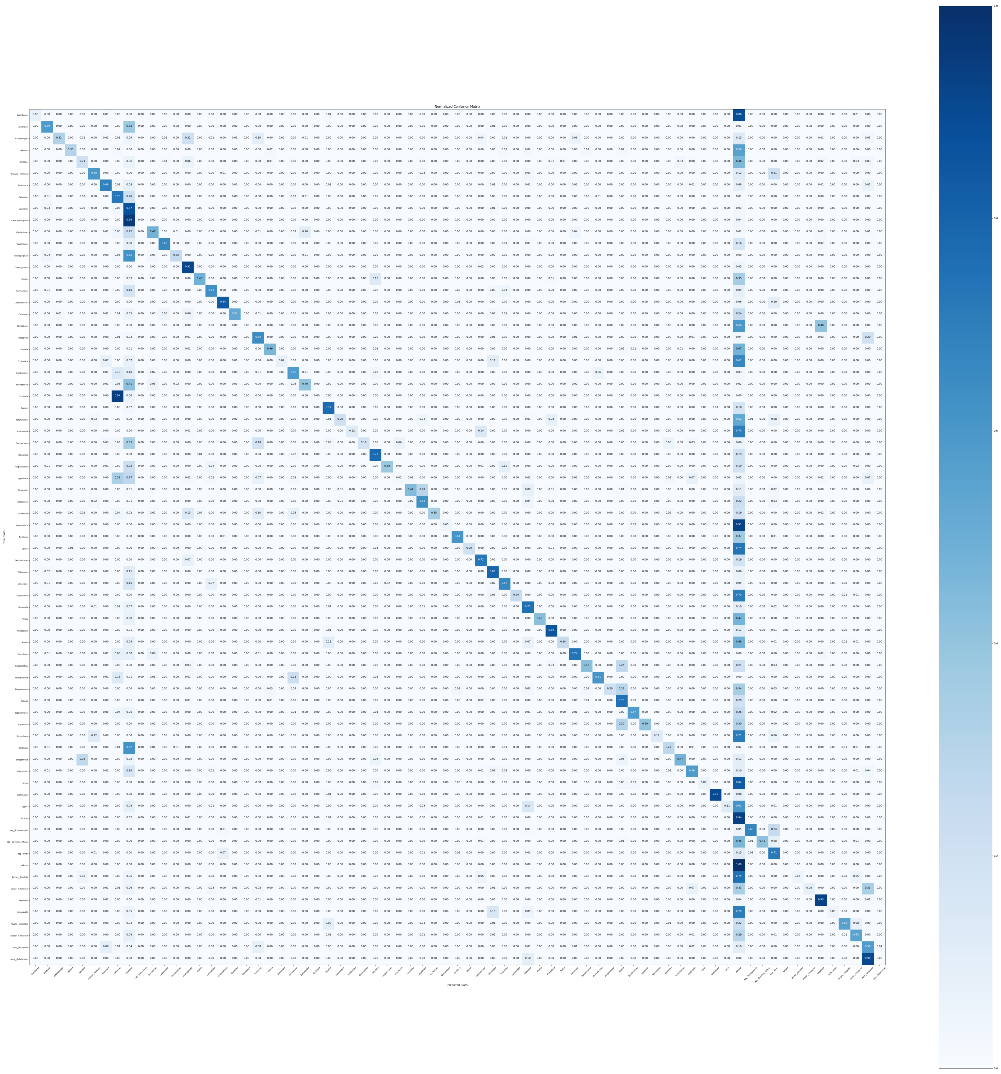


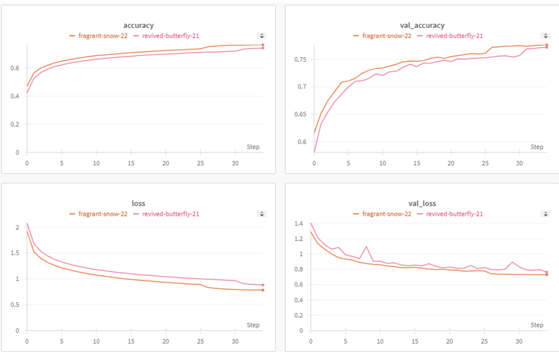
Figure 4.9: Normalized confusion matrix of DenseNet169 using the ZooScan dataset(mb1_all.png on github in the results folder).

4.2.2 Second Benchmark

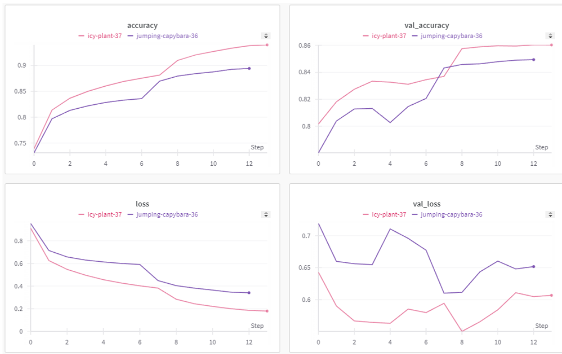
The second benchmark was done in a similar manner as with the Organized dataset, adding two more FC layers, using both 32 and 128 for batch sizes. In addition to this the dataset was limited to maximum 15 000 images per class, to limit the large classes and make the process a bit faster as it now only iterates through 300 000 images instead of 950 000. Table 4.5 shows that both DenseNet and MobileNet had a lower accuracy than in the first benchmark without optimization, however the F1 score has increased by 11 percent for DenseNet and MobileNet 11/14 percent. Which tells us there is less mis-classification. Figure 4.10 displays the accuracy and loss graph and have similar trend as the Organized dataset had on MobileNet; overfitting on the validation and not converging. Looking at each class F1 score (from the best result from table 4.5) there is just one class that have a score of zero, *Zoea Galatheidae*, and some of the classes from the first benchmark that had zero now have a better score and are not mis-classified as *detritus*, among others. This can be seen in the confusion matrix from the best run in figure 4.12. Compared to the confusion matrix from the first benchmark, the mis-classifications are at a much lower rate.

Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
ZooScan(max 15k)	densenet169Opt	32	73	0,74	0,77	0,77	0,60
ZooScan(max 15k)	densenet169Opt	128	73	0,76	0,78	0,77	0,60
ZooScan(max 15k)	mobilenetLOpt	32	73	0,89	0,85	0,84	0,73
ZooScan(max 15k)	mobilenetLOpt	128	73	0,94	0,86	0,86	0,76

Table 4.5: Results from training the ZooScan dataset on optimized DenseNet and MobileNet and different batch sizes.

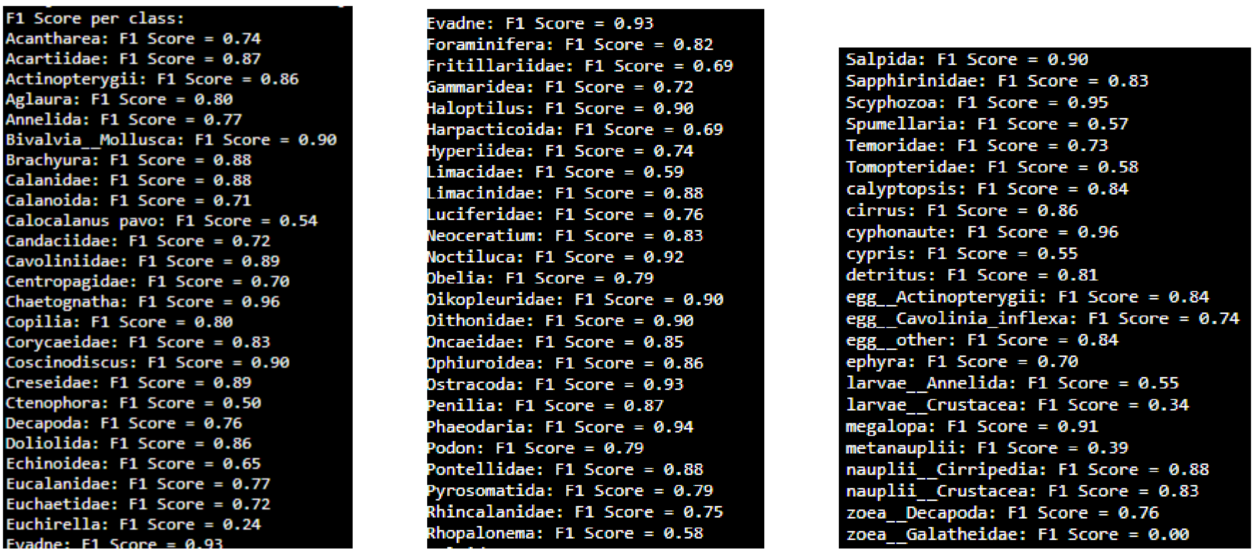


(a) DenseNet169



(b) MobileNetV3Large

Figure 4.10: Metrics during the second benchmark training.



F1 Score per class:
Acantharea: F1 Score = 0.74
Acartiidae: F1 Score = 0.87
Actinopterygii: F1 Score = 0.86
Aglaura: F1 Score = 0.80
Annelida: F1 Score = 0.77
Bivalvia__Mollusca: F1 Score = 0.90
Brachyura: F1 Score = 0.88
Calanidae: F1 Score = 0.88
Calanoida: F1 Score = 0.71
Calocalanus pavo: F1 Score = 0.54
Candaciidae: F1 Score = 0.72
Cavoliniidae: F1 Score = 0.89
Centropagidae: F1 Score = 0.70
Chaetognatha: F1 Score = 0.96
Copilia: F1 Score = 0.80
Corycaeidae: F1 Score = 0.83
Coscinodiscus: F1 Score = 0.90
Creseidae: F1 Score = 0.89
Ctenophora: F1 Score = 0.50
Decapoda: F1 Score = 0.76
Doliolida: F1 Score = 0.86
Echinoidea: F1 Score = 0.65
Eucalanidae: F1 Score = 0.77
Euchaetidae: F1 Score = 0.72
Euchirella: F1 Score = 0.24
Evadne: F1 Score = 0.93

Evadne: F1 Score = 0.93
Foraminifera: F1 Score = 0.82
Fritillariidae: F1 Score = 0.69
Gammaridea: F1 Score = 0.72
Haloptilus: F1 Score = 0.90
Harpacticoida: F1 Score = 0.69
Hyperiidea: F1 Score = 0.74
Limacidae: F1 Score = 0.59
Limacinidae: F1 Score = 0.88
Luciferidae: F1 Score = 0.76
Neoceratium: F1 Score = 0.83
Noctiluca: F1 Score = 0.92
Obelia: F1 Score = 0.79
Oikopleuridae: F1 Score = 0.90
Oithonidae: F1 Score = 0.90
Oncaeidae: F1 Score = 0.85
Ophiuroidea: F1 Score = 0.86
Ostracoda: F1 Score = 0.93
Penilia: F1 Score = 0.87
Phaeodaria: F1 Score = 0.94
Podon: F1 Score = 0.79
Pontellidae: F1 Score = 0.88
Pyrosomatida: F1 Score = 0.79
Rhincalanidae: F1 Score = 0.75
Rhopalonema: F1 Score = 0.58

Salpida: F1 Score = 0.90
Sapphirinidae: F1 Score = 0.83
Scyphozoa: F1 Score = 0.95
Spumellaria: F1 Score = 0.57
Temoridae: F1 Score = 0.73
Tomopteridae: F1 Score = 0.58
calyptopsis: F1 Score = 0.84
cirrus: F1 Score = 0.86
cyphonaute: F1 Score = 0.96
cypris: F1 Score = 0.55
detritus: F1 Score = 0.81
egg__Actinopterygii: F1 Score = 0.84
egg__Cavolinia_inflexa: F1 Score = 0.74
egg__other: F1 Score = 0.84
ephyra: F1 Score = 0.70
larvae__Annelida: F1 Score = 0.55
larvae__Crustacea: F1 Score = 0.34
megalopa: F1 Score = 0.91
metanauplii: F1 Score = 0.39
nauplii__Cirripedia: F1 Score = 0.88
nauplii__Crustacea: F1 Score = 0.83
zoea__Decapoda: F1 Score = 0.76
zoea__Galatheidae: F1 Score = 0.00

Figure 4.11: The F1 score of each class from testing MobileNet.

4.2.3 Third Benchmark

Now the weights created from the "Organized" dataset using the best model from MobileNet (64 batches, four FC layers) was implemented. These weights are trained on similar images (plankton), contrary to the "imagenet" weights that have been trained on various image categories. This was tried on both the full dataset (ZooScan(all)) the 15k max dataset (ZooScan (max 15k)), and only tried on MobileNet as this has given the best results from the previous benchmarks. On the full dataset table 4.6 we see that in the first test, with a batch size of 32, the accuracy is higher than in the first benchmark, but the F1 score is slightly lower. With a batch size of 128 it does much better and both the accuracy and F1 score is higher. In the first benchmark the accuracy was 85% and F1 score 62%. For the max 15k dataset it had almost exactly the same result as in the second benchmark (the training accuracy was one percent lower in this benchmark). In figure 4.13 we see that both datasets have similar trends and more overfitting with the validation. The f1 score from each class in

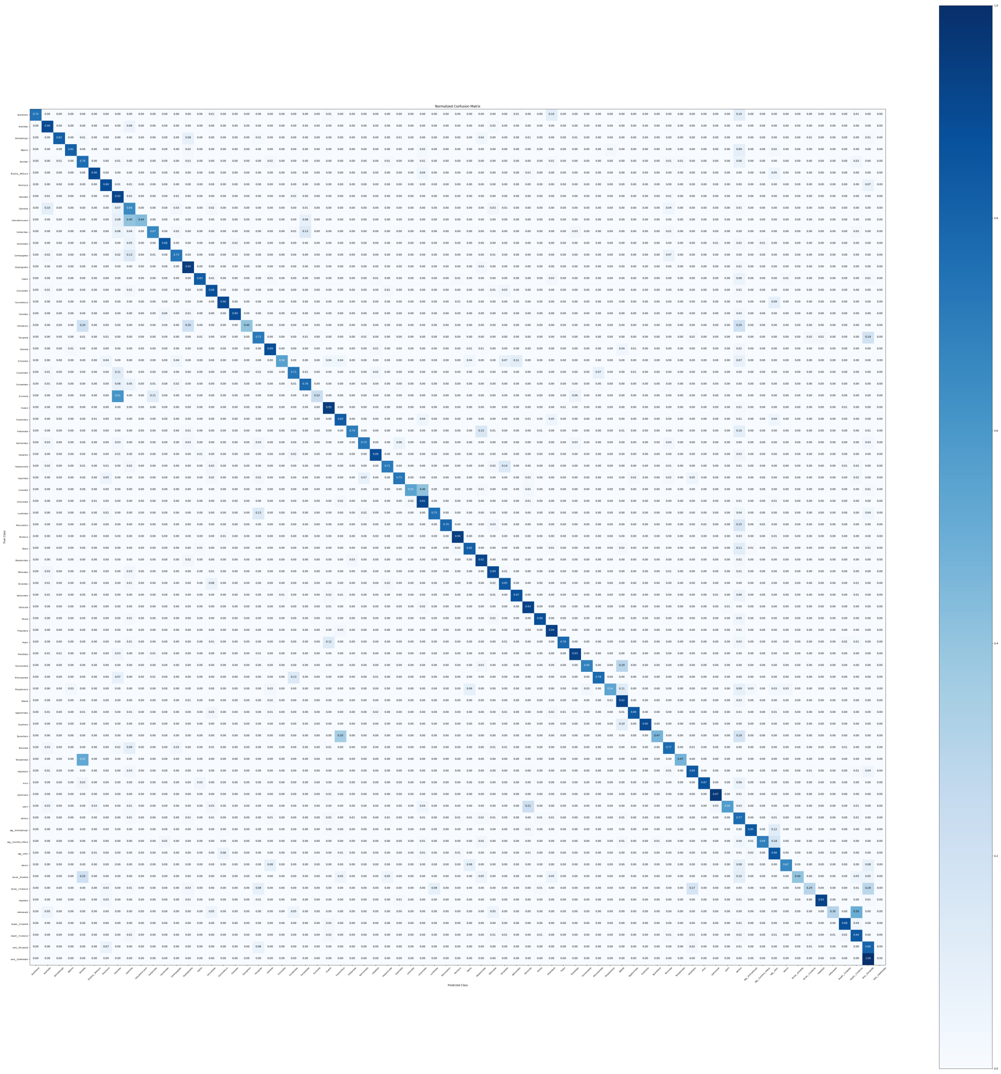
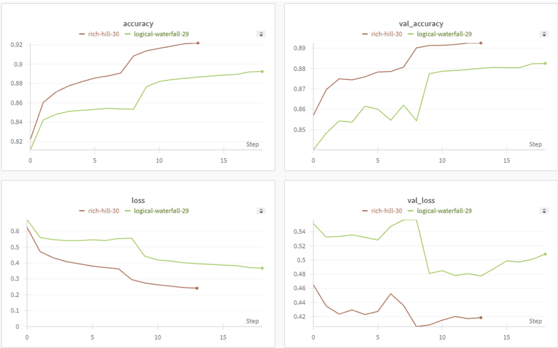


Figure 4.12: Normalized confusion matrix of MobileNet, the best run, using the ZooScan dataset (mb2_15k.png on github in the results folder).

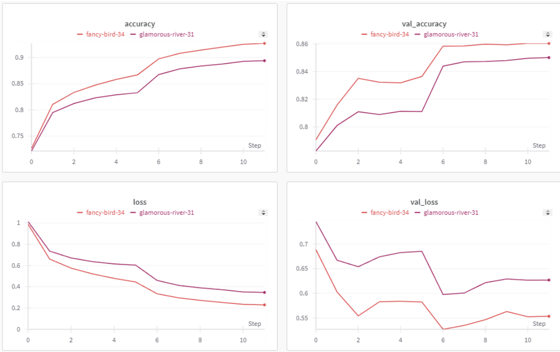
4.14 is a bit different from the second benchmark, with three classes that now have zero F1 score (*Ctenophora*, *Euchirella* and *Zoea Galatheidae*). There also is an improvement between the confusion matrix from the ZooScan(all), figure 4.15, in this benchmark compared to the first benchmark.

Dataset	Method	Batch size	classes	train_acc	val_acc	test_acc	f1(macro)
ZooScan(all)	mobilenetLOpt	32	73	0,89	0,88	0,88	0,60
ZooScan(all)	mobilenetLOpt	128	73	0,92	0,89	0,89	0,71
ZooScan(max 15k)	mobilenetLOpt	32	73	0,89	0,84	0,84	0,73
ZooScan(max 15k)	mobilenetLOpt	128	73	0,93	0,86	0,83	0,76

Table 4.6: Results from training the organized dataset on optimized DenseNet and MobileNet.



(a) Zooscan(all)



(b) ZooScan(max 15k)

Figure 4.13: Metrics during the fourth benchmark training.

```
F1 Score per class:
Acantharea: F1 Score = 0.78
Acartiidae: F1 Score = 0.88
Actinopterygii: F1 Score = 0.87
Aglaura: F1 Score = 0.84
Annelida: F1 Score = 0.74
Bivalvia__Mollusca: F1 Score = 0.90
Brachyura: F1 Score = 0.89
Calanidae: F1 Score = 0.89
Calanoida: F1 Score = 0.71
Calocalanus pavo: F1 Score = 0.52
Candaciidae: F1 Score = 0.74
Cavoliniidae: F1 Score = 0.87
Centropagidae: F1 Score = 0.68
Chaetognatha: F1 Score = 0.96
Copilia: F1 Score = 0.83
Corycaeidae: F1 Score = 0.82
Coscinodiscus: F1 Score = 0.90
Creseidae: F1 Score = 0.86
Ctenophora: F1 Score = 0.00
Decapoda: F1 Score = 0.77
Doliolida: F1 Score = 0.86
Echinoidea: F1 Score = 0.67
Eucalanidae: F1 Score = 0.81
```

```
Euchaetidae: F1 Score = 0.72
Euchirella: F1 Score = 0.00
Evadne: F1 Score = 0.92
Foraminifera: F1 Score = 0.81
Fritillariidae: F1 Score = 0.67
Gammaridea: F1 Score = 0.59
Haloptilus: F1 Score = 0.91
Harpacticoida: F1 Score = 0.72
Hyperiidea: F1 Score = 0.70
Limacidae: F1 Score = 0.70
Limacinidae: F1 Score = 0.89
Luciferidae: F1 Score = 0.80
Neoceratium: F1 Score = 0.87
Noctiluca: F1 Score = 0.92
Obelia: F1 Score = 0.82
Oikopleuridae: F1 Score = 0.90
Oithonidae: F1 Score = 0.90
Oncaeidae: F1 Score = 0.84
Ophiuroidea: F1 Score = 0.87
Ostracoda: F1 Score = 0.93
Penilia: F1 Score = 0.87
Phaeodaria: F1 Score = 0.94
Podon: F1 Score = 0.76
Pontellidae: F1 Score = 0.90
```

```
Pyrosomatida: F1 Score = 0.83
Rhincalanidae: F1 Score = 0.76
Rhopalonema: F1 Score = 0.68
Salpida: F1 Score = 0.91
Sapphirinidae: F1 Score = 0.84
Scyphozoa: F1 Score = 0.90
Spumellaria: F1 Score = 0.79
Temoridae: F1 Score = 0.71
Tomopteridae: F1 Score = 0.62
calyptopsis: F1 Score = 0.84
cirrus: F1 Score = 0.83
cyphonaute: F1 Score = 0.96
cypris: F1 Score = 0.62
detritus: F1 Score = 0.81
egg__Actinopterygii: F1 Score = 0.81
egg__Cavolinia_inflexa: F1 Score = 0.74
egg__other: F1 Score = 0.85
ephyra: F1 Score = 0.60
larvae__Annelida: F1 Score = 0.30
larvae__Crustacea: F1 Score = 0.37
megalopa: F1 Score = 0.93
metanauplii: F1 Score = 0.28
nauplii__Cirripedia: F1 Score = 0.88
nauplii__Crustacea: F1 Score = 0.83
zoea__Decapoda: F1 Score = 0.74
zoea__Galatheidae: F1 Score = 0.00
```

Figure 4.14: The F1 score of each class from running ZooScan(max 15k) on MobileNet (the run with the highest score).

```
F1 Score per class:
Acantharea: F1 Score = 0.66
Acartiidae: F1 Score = 0.83
Actinopterygii: F1 Score = 0.84
Aglaura: F1 Score = 0.77
Annelida: F1 Score = 0.69
Bivalvia__Mollusca: F1 Score = 0.87
Brachyura: F1 Score = 0.88
Calanidae: F1 Score = 0.77
Calanoida: F1 Score = 0.88
Calocalanus pavo: F1 Score = 0.43
Candaciidae: F1 Score = 0.62
Cavoliniidae: F1 Score = 0.86
Centropagidae: F1 Score = 0.53
Chaetognatha: F1 Score = 0.94
Copilia: F1 Score = 0.83
Corycaeidae: F1 Score = 0.81
Coscinodiscus: F1 Score = 0.89
Creseidae: F1 Score = 0.84
Ctenophora: F1 Score = 0.00
Decapoda: F1 Score = 0.75
Doliolida: F1 Score = 0.82
Echinoidea: F1 Score = 0.46
Eucalanidae: F1 Score = 0.76
Euchaetidae: F1 Score = 0.57
```

```
Euchaetidae: F1 Score = 0.57
Euchirella: F1 Score = 0.00
Evadne: F1 Score = 0.88
Foraminifera: F1 Score = 0.75
Fritillariidae: F1 Score = 0.61
Gammaridea: F1 Score = 0.59
Haloptilus: F1 Score = 0.86
Harpacticoida: F1 Score = 0.64
Hyperiidea: F1 Score = 0.70
Limacidae: F1 Score = 0.64
Limacinidae: F1 Score = 0.85
Luciferidae: F1 Score = 0.80
Neoceratium: F1 Score = 0.75
Noctiluca: F1 Score = 0.83
Obelia: F1 Score = 0.73
Oikopleuridae: F1 Score = 0.86
Oithonidae: F1 Score = 0.88
Oncaeidae: F1 Score = 0.83
Ophiuroidea: F1 Score = 0.81
Ostracoda: F1 Score = 0.91
Penilia: F1 Score = 0.78
Phaeodaria: F1 Score = 0.91
Podon: F1 Score = 0.75
Pontellidae: F1 Score = 0.89
Pyrosomatida: F1 Score = 0.85
Rhincalanidae: F1 Score = 0.74
Rhopalonema: F1 Score = 0.57
Salpida: F1 Score = 0.85
Sapphirinidae: F1 Score = 0.82
Scyphozoa: F1 Score = 0.86
```

```
Spumellaria: F1 Score = 0.48
Temoridae: F1 Score = 0.66
Tomopteridae: F1 Score = 0.65
calyptopsis: F1 Score = 0.81
cirrus: F1 Score = 0.57
cyphonaute: F1 Score = 0.95
cypris: F1 Score = 0.52
detritus: F1 Score = 0.94
egg__Actinopterygii: F1 Score = 0.83
egg__Cavolinia_inflexa: F1 Score = 0.73
egg__other: F1 Score = 0.81
ephyra: F1 Score = 0.15
larvae__Annelida: F1 Score = 0.29
larvae__Crustacea: F1 Score = 0.32
megalopa: F1 Score = 0.91
metanauplii: F1 Score = 0.34
nauplii__Cirripedia: F1 Score = 0.84
nauplii__Crustacea: F1 Score = 0.79
zoea__Decapoda: F1 Score = 0.73
zoea__Galatheidae: F1 Score = 0.00
```

Figure 4.15: The F1 score of each class from running ZooScan(all) on MobileNet (the run with the highest score).

4.3 Siamese

From the "Data hacker" github the pairs were created, and it calculates the distance between two images. Figure 4.16 shows an example of one class compared to a random selection of another class. The first one has a low score, meaning they are very likely to be in the same class, while the second one, with a high score, are very likely to be two different classes.

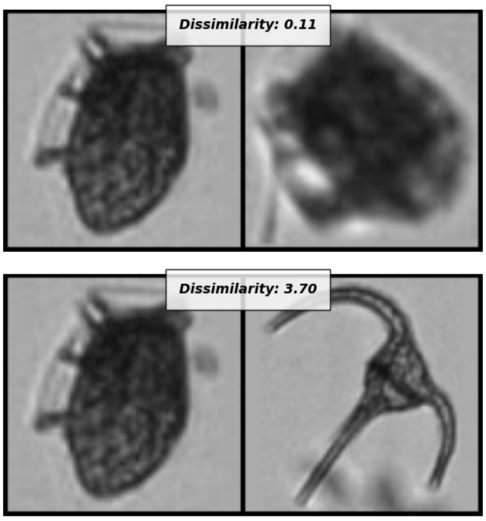


Figure 4.16: Shows the dissimilarity score between random validation images from the Organized dataset. The higher the score, the more dissimilar they are.

The "Siamese MNIST" was easy to reproduce the original results from and got an training accuracy of 97.6% and test accuracy of 96.7%. However to get to work with a dataset with folder structure was not achieved. Figure 4.17 shows the pairs after training and calculating the distance. We see that similar images have a very low score, and dissimilar images have a higher score.

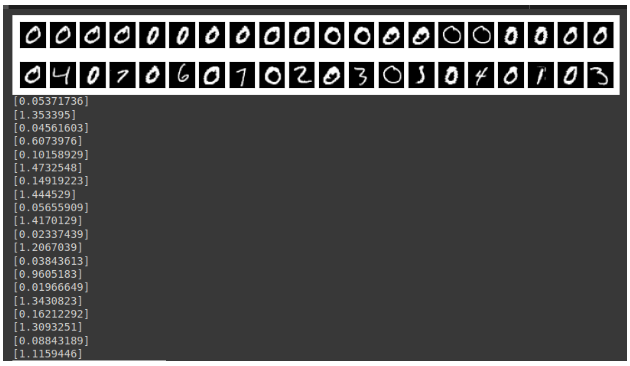


Figure 4.17: Shows pairs and the distance between them, result reproduced from .

Chapter 5

Discussions

For this project two dataset were tested and benchmarked. Each benchmark is altering the training and trying to improve previous results. This chapter will discuss the results for each benchmark and finally compare and conclude the results.

5.1 Organized_v7

The Organized dataset had three benchmarks during this project, and each benchmark will be discussed below.

5.1.1 First Benchmark

As this dataset have been benchmarked in previous studies, the first benchmark in this report was to verify that the results could be reproduced. The results from the thesis [35], the dataset was created from, had identical results using transfer learning for both MobileNet and DenseNet. This created a baseline for the results that are expected from optimizing. For the classes that did well in the confusion matrix, figure 4.2, we can look at their shape in figure 2.9 and see that *appendicularia* and *silicoflagellate* have very unique shapes and can be easy for a model to extract the features from them and not mis-classify them. While *bivalve* and *dinoflagelatte* have less distinct features and other classes are often mis-classified as them.

5.1.2 Second Benchmark

In this test the optimization has been added, with 4 FC layers instead of two as the original test used, and a higher batch size was tested. A table 4.2 displays, DenseNet had a small increase in training accuracy and F1 score. However it did better on the lower batch size, as opposed to MobileNet which showed a slight improvement from iterating through a higher amount of images at a time. MobileNet also shows that it works better using adding more complexity. One distinct difference from the first benchmark is that now the training accuracy is higher than the validation and testing accuracy, meaning it is overfitting and it might be too complex. Reducing input neurons could make the model converge faster and not overfitt. Adding dropout layers might also help, but this is already implemented in both DenseNet and MobileNet. The classes with the lowest F1 score, *bivalve* and *dinoflagelatte*, had a lower score than in the first benchmark, this might be different in another test, as the configuration is non-deterministic (explained in the comparison section below, 5.4), or the features of some of the other classes might be learned better with a more complex model, and features of a class with lower samples and more common features is more difficult to classify. Adding more images of these classes could improve the result.

5.1.3 Third Benchmark

The third benchmark for this dataset was done using weights from the ZooScan dataset. DenseNet did worse than in the second benchmark and the validation accuracy is higher than the training accuracy. This could be due to a too high dropout during training or it could benefit from more epochs to train on. DenseNet have run the full 30 epochs for each test, but as the computational power is limited, it is difficult to run for longer. It also takes 7 hours to complete 30 epochs, and as time is a limit and only one model can be run at a time this was a limitation for this project. MobileNet has, as in the second benchmark, a good score, with training accuracy slightly higher than validation. But again show signs of overfitting and would might benefit from less input neurons.

5.2 ZooScan

This section will cover the results from using the ZooScan dataset, which also had three benchmarks for testing both optimizing and different weights.

5.2.1 First Benchmark

As mention previously, this test was done with all original dataset and network. The accuracy and F1 score was not bad, but there was mis-classifications some classes that had zero F1 score. In table 5.1 the classes that had 0.30 or below in F1 score are shown from MobileNet. An immediate correlation between these classes is that they have a very low image count and are mis-classified as classes with more images. An exception is *spcypozoa*, which had the second lowest image count (86 images) but had a F1 score of 0,89, and had *detritus* as its only false negative. This species has however more unique shapes and will most likely require less images to learn the features compared to some of the other classes that have more similar characteristics to other classes. The classes that had bad results, more discussed in section 5.3, will either require more images, augmentation or be removed and used in networks that require less images to learn features. *Detritus* does have an high F1 score, but it has the larges amount of samples(>500 000), and has a large amount of false positives (mis-classified as other species). The other class with a substantially larger amount of samples, *calanoida* with >200 000 images, is also mis-classified. Limiting the amount of images, as the model seems to learn too many features, would and have a slightly more balanced dataset would likely eliminate the amount of false positives for these two classes.

Name	F1 score	image count
calocalanus pavo	0	236
ctenophora	0	39
echinoidea	0,26	258
euchirella	0	172
gammaridea	0,30	321
spumellaria	0,21	156
cirrus	0,28	297
ephyra	0	108
larvae_annelida	0,10	188
larvae_crustacea	0,26	708
metanauplii	0,16	186
zoea_galatheidae	0	205

Table 5.1: The classes with an F1 score of 30% or lower from the first benchmark.

5.2.2 Second Benchmark

The results from DenseNet are a bit different from the first benchmark. This is expected as the dataset now is limited to a maximum of 15k images per class. The accuracy on both training and validation/testing is lower, however the F1 score is 11% higher. Training it for more epochs would likely improve the accuracy as the classification is better, and the training ran for the full 30 epochs it was set to. MobileNet has a training accuracy and F1 score that has improved from the first benchmark. But as with DenseNet the validation and test accuracy is lower and shows signs of overfitting and is learning the training data too well. Decreasing the input neurons on the FC layers could be a solution here as well.

Table 5.2 shows the updated F1 scores from the lower classes that was achieved with training with a dataset that had a max value of 15k images per class, and using optimization. Now there is only one class with zero F1 score, and the rest of the classes that had low scores now have a much higher score. This tells us that a more generalized learning been achieved and the overall F1 score for the model is higher, the best being 76%, while it was 62% for MobileNet in the first benchmark.

Name	F1 score	image count
calocalanus pavo	0,54	236
ctenophora	0,50	39
echinoidea	0,65	258
euchirella	0,24	172
gammaridea	0,72	321
spumellaria	0,57	156
cirrus	0,86	297
ephyra	0,70	108
larvae_annelida	0,55	188
larvae_crustacea	0,34	708
metanauplii	0,39	186
zoea_galatheidae	0	205

Table 5.2: Updated F1 scores from the classes that had an F1 score of 30% or lower from the first benchmark.

5.2.3 Third Benchmark

In the third and final benchmark only MobileNet is used. It is optimized and the weights from the best result from the Organized dataset. These weights are trained on a similar dataset but, as explained previously, it has less classes and different plankton species with images of a different resolution and that uses FlowCam instead of ZooScan. The results from the ZooScan(all) shows improvement, and in table 5.4 we see the F1 scores, compared to the first benchmark there now are three classes with zero score, the best F1 score from using batch size of 128 got an F1 score of 71%, which is an improvement from the first benchmark and more features from classes with lower samples are extracted.

With ZooScan(max 15k) the results were pretty similar to the second benchmark, even if the result is not better, it proves that the weights from the dataset we have can be used to classify plankton on dataset it has not seen before, and that the dataset has sturdiness. In table 5.3 there also is three classes with zero F1 score, however the F1 score of the model has the same value as the one in the second benchmark, so these weights have extracted features a bit differently and correctly classifying some classes better than with the weights used in the previous test, but struggles more with classes with low image count.

Name	F1 score	image count
calocalanus pavo	0,52	236
ctenophora	0	39
echinoidea	0,67	258
euchirella	0	172
gammaridea	0,59	321
spumellaria	0,79	156
cirrus	0,83	297
ephyra	0,60	108
larvae_annelida	0,30	188
larvae_crustacea	0,37	708
metanauplii	0,28	186
zoea_galatheidae	0	205

Table 5.3: The updated F1 scores on the low score classes on the ZooScan(max 15k) dataset.

Name	F1 score	image count
calocalanus pavo	0,43	236
ctenophora	0	39
echinoidea	0,46	258
euchirella	0	172
gammaridea	0,59	321
spumellaria	0,48	156
cirrus	0,86	297
ephyra	0,15	108
larvae_annelida	0,29	188
larvae_crustacea	0,32	708
metanauplii	0,34	186
zoea_galatheidae	0	205

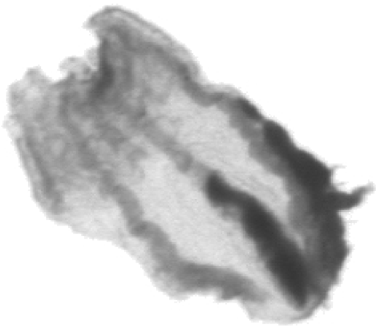
Table 5.4: The updated F1 scores on the low score classes on the ZooScan(all) dataset.

5.3 Low Score Classes

To look further into why some of the classes got a low score from table 5.1, one example is *ctenophora*, seen in figure 5.1, this species has issues with degradation and getting samples can be an issue. An article published in Frontiers "Assessing the Value of a Citizen Science Approach for Ctenophore Identification", [15], states that this specific species is hard to work with and the sample process and preservation of the samples is difficult. Then given low samples and some images with low resolution, this can be difficult to classify. In addition to this we can see on the confusion matrix in figure 4.12 that *Annelida* and *Chaeteognatha* are mis-classified as *ctenophora*. These classes does not look like each other, but in some images where the resolution is lower and their body is twisted, they do have some similar features.

As for *larvea crustacea* it has a higher image count than the other low classes, however it has a lower image count than the too classes that are mis-classified as *larvea crustacea*. *Calyptopsis* and *Zoea Decoapoda* both have false positives as this larvae, they also have a larvae shape to their body, and to fully distinct them from each other more images of the low class, or balance out the images between them could be done to not over-learn the features of the classes with the higher image count.

Metanauplii have some images that are of very low resolution, seen in figure 5.1, which could contribute to the model having a hard time classifying it. Another class, *nauplii crustacea* is mis-classified as *metanauplii*. This class also have low resolution images and in some angles they look alike and could have similar features.


(a) Ctenophora, res: 234x201


(b) Larvae Crustacea, res: 219x247


(c) Metanauplii, res: 54x74

Figure 5.1: Exmaples of groups with a low F1 score.

Zoea Galatheidae, figure 5.2, is the only class to have a zero F1 score on all the benchmarks. It had a low number of images, but if we look at the confusion matrix in figure 4.12 we see that it has an 100% false positive where *Zoea Decoapoda* is mis-classified as it. As they both are larva and have similar shapes, it is likely that the features will be similar. *Zoea Decoapoda* also has the highest image count (6400 images) and it is likely that this is the class the model learns and as *Zoea Galatheidae* only has 205 images this class would need more images to be recognized.

Figure 5.2: Zoea Galatheidae, res: 167x184

5.4 Comparison

Comparing the scores from both datasets on benchmark one and two tells us that adding more FC layers and increasing the batch size did give us a better accuracy and improved the F1 score. MobileNet also had the best performance on both datasets. Looking at the SLR table 2.1 we see that the overall best accuracy we got (training: 98%, test 96%, Flow-Cam) was comparative to the best result from the SLR (accuracy: 97.4%, F1: 96.2% [20]), however the F1 score from this project test was 88%, which means there are a lot more mis-classifications, particularly with the two classes *bivalve* and *dinoflagellate* which respectively got an F1 score of 64% and 49%, while the ten other classes had a high score. For the ZooScan data our best results were training accuracy of 94% and F1 of 76% (from the second benchmark), in the SLR the other studies that used ZooScan had a similar accuracy

(94.6%, [26]). But as previously mention the results from this test had a lower validation accuracy, and there seems to be some overfitting as the model learns the training data very well, but struggles when new validation and test data is introduced.

When the weights used to train the ZooScan dataset was changed from imagenet to the ones created from the best run from the Organized dataset, the results were pretty similar.

For the configuration the model is non-deterministic, so test on the same dataset with the same configuration may vary from each run. This makes it difficult to conclude that with each benchmark the result went up a certain percent, however the trend was positive and the results were improved with the test conducted in this study.

As to why ZooScan produced lower results than from the Organized dataset, FlowCam gives samples with a more stable resolution as the size of objects are limited to a certain size, while ZooScan can have very low or very high resolutions as it can image more various sizes of objects. This can affect the quality of the images in the dataset and make it more difficult to extract features. The Organized dataset also only have 12 classes, while ZooScan have 73, and some of them, as mentioned previously, have very low image count. Fewer classes makes it easier to achieve a higher average score.

5.5 Siamese

Siamese network was started working with, the code from "Data Hacker" gave us results where it could tell the distance between two images, however it was not creating a model to use as a classifier. The results from calculating the distance did seem promising.

With the "Siamese MNIST" code it was running with the MNIST dataset, but that dataset is structured as file that has a reference to where the image is stored and its label, not images in folders. The accuracy for the MNIST dataset is very good, as this is a classifier the goal was to combine these two codes and get a classifier working for our dataset. Unfortunately this work was not completed, but the code so far is added to the repository and can be accessed for potentially working more with. Testing this with the ZooScan dataset would be very interesting to see if the classes that had lower scores would perform better.

Chapter 6

Conclusions

To conclude this paper we take a look at the research questions and whether they have been answered. The first one being: "Will application of optimization on a convolution neural network improve plankton classification?" It is difficult to conclude that this was absolutely possible. The results from previous work on the Organized dataset had an accuracy of 95% during training using MobileNet and 88% with DenseNet. These results were improved to 98% on MobileNet and 89% on DenseNet. The result is not dramatically better, but the accuracy has been improved. However the F1 score was not improved, meaning the actual classification did not improve with this dataset, it only got better at the training dataset. As for the ZooScan dataset the optimization did indeed improve classification. The accuracy for MobileNet and DenseNet was 85% and 78% respectively in the first benchmark. Optimizing it and altering the amount of images on some classes the accuracy was 94% for MobileNet and 89% for DenseNet. The F1 score also went up from 49% to 60% on DenseNet and from 62% to 76% on MobileNet. So for a dataset that already have high results and will need more data curation on certain classes to improve, optimizing the algorithm did not improve the results much. But for a dataset that have low results without much altering of the dataset this did improve the classification.

The second question to answer is: "Can weights learned from models trained on plankton images be used to classify different plankton datasets using transfer learning?" This was tested on both datasets in the third benchmark, using transfer learning with the weights learned from the other dataset. First the weights from the best results from the Organized dataset (second benchmark, MobileNet) was used to train the ZooScan dataset. It did not improve the results from the second benchmark, but the results were almost equal. which proves that the weights learned from the Organized dataset can be used on new datasets to classify different plankton grouups and the result is similar to the state-of-the-art. Doing the opposite and using the weights from the best training from the ZooScan dataset to train the Organized dataset also did not improve the result, but had similar results as the second benchmark. So weights from a small dataset can be used to classify larger datasets, and vice versa.

Siamese network was started working on, however it was not completed and needs more work to classify using the datasets in this project. But the work that was done showed that it can extract features from plankton images and calculate a distance between them.

6.1 Future Work

To further optimize and improve the results for the Optimized dataset, more images of the two classes *bivalve* and *dinoflagellate* could be added to extract more features from them and avoid mis-classification among the other classes. Using different algorithms that have not been much used by the state-of-the-art, like YOLO, would also be very interesting to see if it can capture the features of the classes better.

The ZooScan dataset is a dataset that is already being worked on, but as it has some classes with few images it would also be interesting to try different algorithms. Especially siamese network as that can be a faster and better approach to classification with classes that have fewer images and can possible better capture the features. This was, as seen in the previous chapters, tested, but due to time limitations it was not completed. Had it been finished and used for classification, transfer learning could be implemented to the siamese network and use the weights learned from the Organized dataset could be used for further testing.

Appendix A

Link to Code

The link to the github repository:

https://github.com/katrinalie0303/Optimize-plankton-classification.git

A baseline dataset is provided in the code, the two full datasets are available on request.

Bibliography

[1] Elineau et al. *ZooScanNet: plankton images captured with the ZooScan*. Accessed: 2024-05-08. 2018. DOI: 10.17882/55741. URL: https://www.seanoe.org/data/00446/55741/.

[2] Kanchana Bandara. *UVP6 for plankton imaging and AI-driven image data processing*. Email by request. 2024.

[3] Weights Biases. *What is WB?* Accessed: 2023-11-29. 2023. URL: https://docs.wandb.ai/guides.

[4] Bala Priya C. *Softmax Activation Function: Everything You Need to Know*. Accessed: 2024-05-14. 2023. URL: https://www.pinecone.io/learn/softmax-activation/.

[5] François Chollet. *Transfer learning fine-tuning*. Accessed: 2024-05-14. 2023. URL: https://keras.io/guides/transfer_learning/.

[6] Jan Conradt et al. "Automated Plankton Classification With a Dynamic Optimization and Adaptation Cycle." In: *Frontiers in Marine Science* 9 (2022). ISSN: 2296-7745. DOI: 10.3389/fmars.2022.868420. URL: https://www.frontiersin.org/articles/10.3389/fmars.2022.868420.

[7] Jan Constantin Conradt. *Automated Plankton Image Classification with a Capsule Neural Network*. Master's thesis. 2020.

[8] Jinna Cui et al. "Texture and Shape Information Fusion of Convolutional Neural Network for Plankton Image Classification." In: *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*. 2018, pp. 1–5. DOI: 10.1109/OCEANSKOBE.2018.8559156.

[9] Vyacheslav Efimov. *Understanding Deep Learning Optimizers: Momentum, AdaGrad, RMSProp Adam*. Accessed: 2024-05-14. 2023. URL: https://towardsdatascience.com/understanding-deep-learning-optimizers-momentum-adagrad-rmsprop-adam-e311e377e9c2.

[10] PhD Everton Gomede. *DenseNet: Densely Connected Convolutional Neural Networks*. Accessed: 2023-12-03. 2023. URL: https://medium.com/@evertongomede/densenet-densely-connected-convolutional-neural-networks-0fd219379138.

[11] Gaby Gorsky et al. "Digital zooplankton image analysis using the ZooScan integrated system." In: *Journal of Plankton Research* 32.3 (Mar. 2010), pp. 285–303. ISSN: 0142-7873. DOI: 10.1093/plankt/fbp124. eprint: https://academic.oup.com/plankt/article-pdf/32/3/285/4394627/fbp124.pdf. URL: https://doi.org/10.1093/plankt/fbp124.

[12] Adam T. Greer et al. "In situ imaging across ecosystems to resolve the fine-scale oceanographic drivers of a globally significant planktonic grazer." In: *Limnology and Oceanography* 68.1 (2023), pp. 192–207. DOI: https://doi.org/10.1002/lno.12259. eprint: https://aslopubs.onlinelibrary.wiley.com/doi/pdf/10.1002/lno.12259. URL: https://aslopubs.onlinelibrary.wiley.com/doi/abs/10.1002/lno.12259.

[13] Andrew G. Howard et al. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. 2017. arXiv: 1704.04861 [cs.CV].

[14] Purva Huilgol. *Accuracy vs. F1-Score*. Accessed: 2023-11-30. 2019. URL: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2.

[15] Ellie Johansen et al. "Assessing the Value of a Citizen Science Approach for Ctenophore Identification." In: *Frontiers in Marine Science* 8 (2021). ISSN: 2296-7745. DOI: 10.3389/fmars.2021.772851. URL: https://www.frontiersin.org/articles/10.3389/fmars.2021.772851.

[16] M. I. Jordan and T. M. Mitchell. "Machine learning: Trends, perspectives, and prospects." In: *Science* 349.6245 (2015), pp. 255–260. DOI: 10.1126/science.aaa8415. eprint: https://www.science.org/doi/pdf/10.1126/science.aaa8415. URL: https://www.science.org/doi/abs/10.1126/science.aaa8415.

[17] jupyter.org. *Project Jupyter Documentation*. Accessed: 2023-11-29. 2023. URL: https://docs.jupyter.org/en/latest/.

[18] keras.io. *About Keras 3*. Accessed: 2023-11-29. 2023. URL: https://keras.io/about/.

[19] keras.io. *Keras Applications*. Accessed: 2023-11-29. 2023. URL: https://keras.io/api/applications/.

[20] Thomas Kerr et al. "Collaborative Deep Learning Models to Handle Class Imbalance in FlowCam Plankton Imagery." In: *IEEE Access* 8 (2020), pp. 170013–170032. DOI: 10.1109/ACCESS.2020.3022242.

[21] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[22] Bharath Krishnamurthy. *An Introduction to the ReLU Activation Function*. Accessed: 2024-05-14. 2024. URL: https://builtin.com/machine-learning/relu-activation-function.

[23] Dr. Marta Moyano. *THRESHOLDS: Disentangling the effects of climate-driven processes on North Sea herring recruitment through physiological thresholds*. project proposal. Jan. 2017.

[24] Rinki Nag. *A Comprehensive Guide to Siamese Neural Networks*. Accessed: 2023-12-03. 2022. URL: https://medium.com/@rinkinag24/a-comprehensive-guide-to-siamese-neural-networks-3358658c0513.

[25] Sushant Patrikar. *Batch, Mini Batch Stochastic Gradient Descent*. Accessed: 2024-05-10. 2019. URL: https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a.

[26] Rene-Marcel Plonus et al. "Automatic plankton image classification—Can capsules and filters help cope with data set shift?" In: *Limnology and Oceanography: Methods* 19.3 (2021), pp. 176–195. DOI: https://doi.org/10.1002/lom3.10413. eprint: https://aslopubs.onlinelibrary.wiley.com/doi/pdf/10.1002/lom3.10413. URL: https://aslopubs.onlinelibrary.wiley.com/doi/abs/10.1002/lom3.10413.

[27] Waseem Rawat and Zenghui Wang. "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review." In: *Neural Computation* 29.9 (2017), pp. 2352–2449. DOI: 10.1162/neco_a_00990.

[28] Pablo Ruiz. *Understanding and visualizing DenseNets*. Accessed: 2023-12-03. 2018. URL: https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a.

[29] Moritz S. Schmid et al. "Prey and predator overlap at the edge of a mesoscale eddy: fine-scale, in-situ distributions to inform our understanding of oceanographic processes." In: *Scientific Reports* 10.1 (Jan. 2020), p. 921. ISSN: 2045-2322. DOI: 10.1038/s41598-020-57879-x. URL: https://doi.org/10.1038/s41598-020-57879-x.

[30] sciencelearn.org. *Plankton*. Accessed: 2023-12-18. 2009. URL: https://www.sciencelearn.org.nz/resources/146-plankton.

[31] David Sosa-Trejo et al. "Vision-based techniques for automatic marine plankton classification." In: *Artificial Intelligence Review* 56 (2023). ISSN: 1573-7462. DOI: 10.1007/s10462-023-10456-w. URL: https://doi.org/10.1007/s10462-023-10456-w.

[32] Kelsey Swieca et al. "Changing with the tides: Fine-scale larval fish prey availability and predation pressure near a tidally modulated river plume." In: *Marine Ecology Progress Series* 650 (Sept. 2020). DOI: 10.3354/meps13367.

[33] YOKOGAWA Fluid Imaging Technologies. *Particle Size and Shape Analysis Using Flow Imaging Microscopy*. 2023. URL: https://www.fluidimaging.com/applications/how-it-works (visited on 11/26/2023).

[34] YOKOGAWA Fluid Imaging Technologies. *The Ultimate Guide to Flow Imaging Microscopy*. https://www.scimed.com.sg/wp-content/uploads/2020/05/Flow-Imaging-Microscopy-General-Ebook-Yokogawa.pdf. Accessed: 2023-11-26.

[35] Gabriel Torland, Kim Andre Svalde, and Andreas Primstad. *Towards Better Plankton Classification*. Bachelor thesis. 2023.

[36] Vishal Yathish. *Loss Functions and Their Use In Neural Networks*. Accessed: 2023-12-07. 2022. URL: https://towardsdatascience.com/loss-functions-and-their-use-in-neural-networks-a470e703f1e9.