



TÉCNICO
LISBOA



Detection of Swimmers in Open Water using Shipborne Vision Sensors

Franziska Stefanie Auer

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Alexandre José Malheiro Bernardino
Lukasz Dzianach, Callboats

Examination Committee

Chairperson: Prof. João Manuel de Freitas Xavier
Supervisor: Prof. Alexandre José Malheiro Bernardino
Member of the Committee: Atabak Dehban

July 2023

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the *Universidade de Lisboa*.

Acknowledgments

I would like to express my gratitude to my supervisors, Professor Alexandre Bernardino and Lukasz Dzianach. They have provided me with expertise and motivation, as well as feedback and support throughout this dissertation project. Their guidance, insight, and sharing of knowledge were vital to the completion of this dissertation.

I would also like to acknowledge the crucial support from my friends who agreed to be photographed to create the dataset for this thesis. Without their time and motivation this thesis would not have been possible.

Most important, I would like to extend my gratitude to my parents for their support throughout my education, as well as their encouragement of my career choice.

Last but not least, to all my friends and colleagues who have been there for me during the good and bad times of my life. You have helped me grow as a person and have been an inspiration to me.

Thank you.

Abstract

When developing autonomous ferries, safety must be a priority. Most times a ferry will not be alone in the water so it will encounter other objects or even humans. This thesis aims to develop a model that detects humans in water to increase the operational safety of an autonomous ferry. The images used are taken from an RGB camera aboard the ferry. The method used in this thesis is object detection which is based on Machine Learning (ML). In this task, the object to be detected is a swimmer. Two different network architectures will be used to detect the swimmer, namely Faster R-CNN and YOLOv8. These models will be trained by using transfer learning with two existing datasets and a newly created dataset that represents exactly the viewing angle of the ferry. Additionally, the newly created dataset features more images in difficult situations like during sunset, with (partially) occluded swimmers and people snorkelling under the water surface. To obtain good detection results, a temporal filtering rule was implemented: Detections of 10 images are combined to obtain a single window detection. Our results show that with YOLOv8x plus the use of windows, it is possible to have less than one False Positive (FP) per season while still assuring that a potential swimmer in front of the boat will be detected.

Keywords

Machine Learning; Human Detection; Swimming; Small Dataset;

Resumo

Ao desenvolver ferries autónomos, a segurança deve ser uma prioridade. Na maioria das vezes, um ferry não estará sozinho na água pelo que encontrará outros objetos ou mesmo humanos enquanto opera. Esta tese visa desenvolver um modelo que detete humanos na água para aumentar a segurança operacional de um ferry autónomo. As imagens utilizadas são tiradas de câmaras (RGB e IR) instaladas a bordo do ferry. O método utilizado nesta tese é a detecção de objectos com base na aprendizagem mecânica (ML). Nesta tarefa o objeto a detetar é um nadador. Três arquiteturas de rede diferentes serão utilizadas para detetar o nadador, nomeadamente a Faster R-CNN e a YOLO. Estes modelos serão treinados usando a aprendizagem por transferência com dois conjuntos de dados existentes e um conjunto de dados recentemente criado que representa exatamente o ângulo de visão do ferry. Além disso, o conjunto de dados recém-criado apresenta mais imagens em situações difíceis, como durante o pôr do sol, com nadadores (parcialmente) ocultos e pessoas a fazer snorkelling sob a superfície da água. Para obter bons resultados de detecção, foi implementada uma regra de filtragem temporal: As detecções de 10 imagens são combinadas para obter uma única janela de detecção. Os nossos resultados mostram que com o YOLOv8x e a utilização de janelas, é possível ter menos de um Falso Positivo (FP) por época de operação e, ao mesmo tempo, garantir que um potencial nadador em frente ao barco seja detectado.

Palavras Chave

Aprendizagem Máquina; Detecção de Humanos; Nadadores;ss Dados Escassos;

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Problem definition	3
1.3	Challenges	4
1.4	Aim and objective	5
1.5	Contribution	6
2	Background	7
2.1	Object Detection	8
2.1.1	Backbone Network	8
2.1.2	Two-stage Detector	8
2.1.3	One-stage Detector	9
2.2	Dataset split	10
2.3	Evaluation	10
2.3.1	Log Average Miss-Rate	10
2.3.2	Intersection Over Union	11
2.3.3	Average Precision	12
2.4	Fusion of detections	12
3	State-of-the-Art	15
3.1	Detecting people on land overview	16
3.1.1	Pedestrian detection RGB camera	17
3.1.2	Pedestrian detection using thermal and Infrared (IR) cameras	17
3.2	Existing water-based datasets	18
3.2.1	Detecting people in water from Unmanned Aerial Vehicle (UAV)	18
3.2.2	Working with small datasets	19
3.3	Variance in existing datasets	21
3.4	Literature Gaps	22

4	Methodology	23
4.1	Algorithms	24
4.1.1	Faster Region-Based Convolutional Neural Network (R-CNN)	24
4.1.2	You Only Look Once (YOLO)	25
4.2	Swimmer detection model	26
4.2.1	Data augmentation	26
4.2.2	Training	27
4.2.2.A	Transfer Learning	27
4.2.2.B	Cross-validation	27
4.2.3	Evaluation	28
4.2.3.A	Object detection performance	28
4.2.3.B	Detection Speed	28
5	Preliminary Experiments	29
5.1	Influence of distance between camera and swimmer	30
5.2	Influence weather	31
5.3	Influence variety of swimmers	33
5.4	Influence amount of people in the image	36
6	Dataset creation	37
6.1	Cameras	38
6.1.1	RGB Camera	38
6.1.2	Thermal Camera	39
6.1.3	iPhone Camera	39
6.2	Annotation Method	39
6.3	New dataset	40
6.3.1	Dataset size	40
6.3.2	Example Images	41
7	Results	45
7.1	Dataset preparation	46
7.1.1	Cross-Validation dataset	46
7.1.2	Dataset split	49
7.1.3	Data augmentation	51
7.2	Faster R-CNN	51
7.2.1	Optimizing Faster R-CNN	51
7.2.2	Performance on test set	51
7.3	YOLO	53

7.3.1	YOLOv8n	54
7.3.2	YOLOv8x	55
7.3.3	Comparison YOLOv8n and YOLOv8x	56
7.3.4	Improving performance on test set	56
7.4	Comparison Faster R-CNN and YOLOv8	57
7.4.1	Comparison on the test set	58
7.4.2	Comparison on images from SeaDronesSee	59
7.5	Probability False Negative (FN) and False Positive (FP) per season	61
7.6	Implementation model	65
8	Conclusion	67
8.1	Outcome	68
8.2	Future work	69
	Bibliography	71
A	Appendix A	76
A.1	Proofs Generalisation	76
A.1.1	Floater vs Swimmer	76
A.1.2	Object in the image	77
A.1.3	Animal in the image	78
A.2	Detection results in sequence	79
A.2.1	Cross-Validation	79
A.2.2	YOLOv8n	80
A.2.3	YOLOv8x	82
A.2.4	Faster R-CNN	83

List of Figures

1.1	Example picture from that dataset of [Lygouras et al., 2019]	2
1.2	Electrical ferry from Callboats	3
1.3	Person occluded by swell [Lygouras et al., 2019]	5
1.4	Hand reflected by water surface [Images, 2020]	5
1.5	Sunlight reflected by water surface [Todabasi, 2022]	5
2.1	Basic architecture of a two-stage detector [Jiao et al., 2019]	9
2.2	Basic architecture of a one-stage detector [Jiao et al., 2019]	9
2.3	Intersection Over Union (IoU) formula; adapted of [Koech, 2020]	11
2.4	Precision-recall curve example; adapted of [Minaee et al., 2022]	11
3.1	Example detection results [Zhang et al., 2020]	17
3.2	Combining RGB and Thermal Images	17
3.3	Usage of IR cameras	17
3.4	Categories in [Varga et al., 2022]	18
3.5	Zoomed in detection results [Cafarelli et al., 2022]	18
3.6	Example edge detector [Leira et al., 2015]	19
3.7	Examples Swimmer Dataset	19
3.8	Augmentation examples [Jacquelin et al., 2022]	20
4.1	Region Proposal Network (RPN) [Ren et al., 2015]	24
4.2	Region Proposal Network (RPN) [Ren et al., 2015]	25
4.3	YOLOv8 architecture [Terven and Cordova-Esparza, 2023]	26
5.1	Swimmer not detected	30
5.2	Swimmer detected as several swimmers	30
5.3	Swimmers not detected	31
5.4	One swimmer detected	31

5.5 Swimmer occluded	31
5.6 One swimmer detected	31
5.7 One swimmer detected	32
5.8 Two out of three detected	32
5.9 Reflection as part of swimmer	32
5.10 Reflection as separate swimmer	32
5.11 Reflection of person on boat	33
5.12 Reflection of person on boat	33
5.13 Correct detection	33
5.14 Reflection of the sun detected as a swimmer	33
5.15 Swimmer with brown skin	34
5.16 Swimmer with black skin	34
5.17 Single child	35
5.18 Group of kids	35
5.19 Person detected too big	35
5.20 Only arms	35
6.1 Beach (image recording location)	38
6.2 Helsinki (ferry operation location)	38
6.3 RGB Camera [Inc, 2021]	39
6.4 Thermal Camera [LLC, 2021]	39
6.5 Examples of swimmers during sunset	41
6.6 Examples of swimmers partially occluded	41
6.7 Examples of swimmers fully occluded	41
6.8 Examples of swimmers without a visible head	42
6.9 Examples of a black swimmer during sunset	42
6.10 Examples of images with one person	43
6.11 Examples of images with two people	43
6.12 Images taken with a thermal camera	43
7.1 Dangerous missed detection	47
7.2 Swimmers on the left not detected	47
7.3 Sunset recording 1	48
7.4 Sunset recoding 2	48
7.5 Sunset recording 3	48
7.6 Correct detection in Sunset 3	48

7.7	All correct in Sunset 2	48
7.8	Example Dataset Split SeaDronesSee [Varga et al., 2022]	49
7.9	Example FNs during sunset	52
7.10	Example dangerous FN	52
7.11	Example FN far from camera	53
7.12	Only dangerous FN close to camera	53
7.13	One person as two	54
7.14	One false positive	54
7.15	Two false positives	54
7.16	One false positive	55
7.17	Only one foot detected	55
7.18	Left person FN	56
7.19	Two FNs	56
7.20	Right person FN	56
7.21	Lowest confidence score for a True Positive (TP)	57
7.22	Several FPs with a low confidence score	57
7.23	Full sequence detected correctly	60
7.24	FN Faster R-CNN	60
7.25	FN YOLOv8x	60
7.26	Scene with group of swimmers	61
7.27	Scene with two swimmers and a boat	61
A.1	Training floaters	77
A.2	Example far away false positive	77
A.3	Training swimmers	77
A.4	Example far away false positive	77
A.5	Swimmer and Object detected as one	78
A.6	Detections on shore	78
A.7	Animal and swimmer detected	78
A.8	Animal detected but not swimmer	78
A.9	Image 163953 071	79
A.10	Image 163954 135	79
A.11	Image 163955 227	79
A.12	Image 163956 319	79
A.13	Image 163957 412	79
A.14	Image 163958 477	79

A.15 Image 163959 574	80
A.16 Image 164000 648	80
A.17 Image 164001 742	80
A.18 Image 164002 813	80
A.19 Image 6482	80
A.20 Image 6484	80
A.21 Image 6489	81
A.22 Image 6494	81
A.23 Image 6495	81
A.24 Image 6498	81
A.25 Image 6505	81
A.26 Image 6507	81
A.27 Image 6509	82
A.28 Image 6511	82
A.29 Image 183539 481	82
A.30 Image 183540 556	82
A.31 Image 183539 481	82
A.32 Image 183542 725	82
A.33 Image 183543 828	83
A.34 Image 183544 907	83
A.35 Image 183545 983	83
A.36 Image 183547 083	83
A.37 Image 183548 171	83
A.38 Image 183549 274	83
A.39 Image 2104	83
A.40 Image 2105	83
A.41 Image 2106	84
A.42 Image 2107	84
A.43 Image 2108	84
A.44 Image 2109	84
A.45 Image 2110	84
A.46 Image 2112	84
A.47 Image 2113	84
A.48 Image 2114	84

List of Tables

2.1	OR rule	12
3.1	Overview Research Papers	16
3.2	Overview Datasets	16
3.3	Amount of images per category in the SeaDronesSee dataset	21
3.4	Amount of images per category in the Swimmer Dataset	21
5.1	Influence amount of people	36
6.1	Recorded images per camera	40
6.2	Recorded images per recording	40
6.3	Overview images per category	41
7.1	Overview images used for final training	46
7.2	Comparison Performance Faster RCNN YOLO	58
7.3	Comparison Results on Test Set	59
7.4	Probability for a window detection results - test set	63
7.5	Probability for a window detection results - SeaDronesSee examples	64

Acronyms

Adam	Adaptive Moment Estimation
AP	Average Precision
CNN	Convolutional Neural Network
CV	Computer Vision
CVAT	Computer Vision Annotation Tool
DL	Deep Learning
EER	Equal Error Rate
FN	False Negative
FP	False Positive
FPPI	False Positives Per Image
GT	Ground Truth
IoU	Intersection Over Union
IR	Infrared
LAMR	Log Average Miss-Rate
mAP	mean Average Precision
ML	Machine Learning
MR	Miss Rate
NMS	Non-Maximum Suppression
P	Precision
PO	Predicted Object
R	Recall
R-CNN	Region-Based Convolutional Neural Network
RGB	Red-Green-Blue

RMSprop	Root Mean Squared Propagation
RoI	Region of Interest
RPN	Region Proposal Network
SAR	Search and Rescue
SGD	Stochastic Gradient Descent
TN	True Negative
TP	True Positive
UAV	Unmanned Aerial Vehicle
YOLO	You Only Look Once

1

Introduction

Contents

1.1 Motivation	2
1.2 Problem definition	3
1.3 Challenges	4
1.4 Aim and objective	5
1.5 Contribution	6

Safety is of utmost importance in the development of autonomous ferries. Given that ferries typically operate in shared waterways, encounters with other vessels or even humans are likely. Therefore, ensuring the safe navigation and interaction of autonomous ferries with their surroundings is crucial.

This master thesis aims to be part of the development of autonomous ferries by increasing operational safety. This is done by creating a model that automatically detects swimmers in water. To set up this model, a dataset of swimmers seen from the camera on board the ferry is formed and then an algorithm is trained on this dataset.

1.1 Motivation

When aiming to build autonomous ferries, safety is an important aspect. Between 2014 and 2020 7,501 marine casualties and incidents in waters of EU Member States involved people (either injured or dead) [EMSA, 2021]. This high number shows how important safety aspects for ships are.

Chances are very low that the ferry will be completely alone in the water. Most times it will encounter other ships. Several research ([Shin et al., 2020], [Prasad et al., 2017], [Benderius et al., 2021]) has already been done on automatically detecting these ships. However, it is always possible that there are humans in the water. Even in a harbour where people are forbidden to swim, no one can guarantee that there are no swimmers.

When a human is swimming in water, it is normally not possible to see the full body of that person due to reflections and diffraction of the water. Most times only the head and one or two arms will be visible. Figure 1.1 shows an example of a person swimming to visualize the commonly noticeable body parts of a swimmer.



Figure 1.1: Example picture from that dataset of [Lygouras et al., 2019]

Every ship nowadays has a captain (and crew) on board. The captain or one of the officers is always scanning the water when the ship is moving to check for anything that might be in the water. Even though these people are not searching for a swimmer in front of the ship, they would recognize one. To allow any autonomous driving, the machine has to be at least as good as a human to be accepted by society. Consequently, it is essential that an autonomous ferry would recognize not only other ships but also humans in front of it.

1.2 Problem definition

Object detection is a computer vision problem that aims to find objects in an image and correctly classify them. The basic inputs needed are at least one image and an algorithm to find the object in the image.

In this master thesis, the object to be found is a swimmer. The thesis is developed in cooperation with the company [Callboats](#). They operate a ferry in Helsinki, Finland. The boat is shown in Figure 1.2. The work in this thesis is based on this ferry but can potentially be transferred to any ferry operating under similar conditions.



Figure 1.2: Electrical ferry from [Callboats](#)

The ferry operates in an environment where it is unlikely to find people swimming in the water. It is assumed that, at any given time, a maximum of two individuals may be present in the water. The boat has an Red-Green-Blue (RGB) camera and a thermal camera installed which will be used to create a new dataset of swimmers from the perspective of the ferry.

The environment, in which the ferry is navigating, is limited to calm open water like urban waterways, rivers, or lakes. This, on first sight, simple environment comes with several challenges: reflections and diffraction on the water surface, waves, that might occlude the swimmer, as well as sunlight reflecting on the water surface can lead to camera sensor saturation.

As stated in [Jacquelin et al., 2022] most public datasets do not feature people in water. Section 3.2 will focus on the already existing datasets that try to fill this gap. However, there is no dataset available for this specific scenario. Therefore, a new dataset is being created as part of this thesis.

The detection of humans in the newly created dataset will be done by using Machine Learning (ML) techniques which are able to cope with big amounts of data. The performance of different ML algorithms on the newly created dataset will be compared to find out which one works best in this specific scenario.

The goal for the ML algorithms' performance is based on the travelling conditions of the ferry:

- Immediate ferry stop if a swimmer is in close proximity.
- Maximum of one unnecessary immediate stop per season.
- Zero false negatives near the ferry.
- Final detection results rely on a sequence of images, considering consecutive image scenarios.

1.3 Challenges

The problem presented in Section 1.2 indicates several challenges. Current state-of-the-art algorithms do not detect perfectly every human in every potential image (see results in Table 3.1). Hence it is not yet possible to guarantee that every swimmer will be detected. Additionally, the ocean is a challenging environment due to its reflections and diffraction on the water surface, waves, that might occlude the swimmer, and sunlight reflections on the water surface, that might saturate the camera sensor.

As shown in Figure 1.3 people in water create several challenges based on the occlusion of most body parts as well as the constant environmental changes due to waves. In calm water situations, the real body part of the human might be reflected almost like a mirror by the water surface as shown in Figure 1.4. The difficulty for a detector is to decide which part of the image is the human and which part is the reflection.



Figure 1.3: Person occluded by swell [Lygouras et al., 2019]



Figure 1.4: Hand reflected by water surface [Images, 2020]



Figure 1.5: Sunlight reflected by water surface [Todabasi, 2022]

On a bright day, sunlight will be reflected by the water's surface. As Figure 1.5 shows this reflected sunlight is very bright and may over-saturate the camera sensor making it difficult to see anything else in the image.

1.4 Aim and objective

The aim of this thesis is to create a model to detect swimmers in open water. Due to limited data available, a new dataset of swimmers based on the perspective of a ferry will be created. This dataset will be evaluated by using 2 different ML algorithms to determine which model performs the best for the given situation. The model has to be accurate and work in real time.

1.5 Contribution

The practical contribution of this thesis is a fine-tuned model that is fast enough to detect potential humans in water while the ferry is moving. This will make autonomous driving on the water surface safer in the future.

This model includes a filtering technique specifically designed to address the challenges outlined in Section 1.3. The filter reduces the impact of individual images on the final detection result of the model, through the implementation of various decision rules.

Additionally, the company [Callboats](#) will be provided with a dataset featuring humans in water based on the ferry's perspective. As well as the assessment of the performance of a swimmer detection model which will help [Callboats](#) determine if this technology works as intended. Both contribute to the social acceptance of an autonomous ferry.

2

Background

Contents

2.1 Object Detection	8
2.2 Dataset split	10
2.3 Evaluation	10
2.4 Fusion of detections	12

Traditional object detection as part of Computer Vision (CV) is based on handcrafted features. These models are cannot optimize the features extracted from the images and were therefore replaced by more powerful Deep Learning (DL) techniques. The networks used for this master thesis are all Convolutional Neural Network (CNN) type networks due to their good performance in object detection tasks [Zhao et al., 2019].

2.1 Object Detection

The goal of object detection is to locate an object in an image and classify it correctly. To indicate where the object is in the image, a bounding box is used.

Originally object detection was based on hand-crafted features (edges and corners inside the image) to detect objects. With the introduction of CNN algorithms more complex features are used as each pixel of an image is represented by a feature description. The usage of these more complex features improved the performance of the detectors [Janai et al., 2020].

Nowadays there exist two object detectors: "Two-stage detectors" and "One-stage detectors". Two-stage object detectors propose first a region where an object could be and in the second stage what object is in this region. One-stage object detectors do the full process in one step [Jiao et al., 2019].

2.1.1 Backbone Network

Most modern object detectors use a backbone network. The backbone network is a basic feature extractor. It uses images as input and outputs a feature map. As shown in Figure 2.1 the backbone network consists mainly of convolution layers that down-sample the image to extract the features.

A backbone network is trained through supervised learning with a labeled dataset. The network is initialized with random weights, and iterative training involves forward and backward propagation, adjusting parameters to optimize predictions.

Due to its ability to extract features, which is very important to obtain good results, a backbone network can be an adequate tool to improve the performance of an object detector [Jiao et al., 2019].

2.1.2 Two-stage Detector

Two-stage detectors first suggest candidate object bounding boxes. Then what class of object could be inside this bounding box.

In the first stage, proposal generation, the algorithm proposes a region where an object could be in the image without proposing a category of the image yet. The image passes through several convolution layers which extract features. Proceeding from the extracted features the algorithm finds the contours

of objects. Based on these contours, the Region Proposal Network (RPN) proposes regions where the object could be [Bappy and Roy-Chowdhury, 2016].

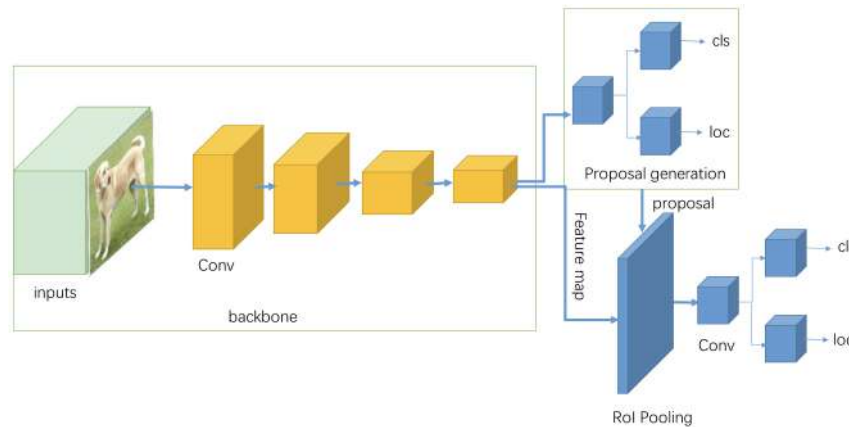


Figure 2.1: Basic architecture of a two-stage detector [Jiao et al., 2019]

In the second stage, a Region of Interest (RoI) Pooling operation combines the features extracted by the backbone with the proposed region. After another convolution layer, the object is classified (cls) and its bounding box/location is given (loc). Figure 2.1 shows the basic architecture of a two-stage detector, including the backbone network [Jiao et al., 2019].

Two-stage detectors usually have high accuracy on where the object is in an image and what object it is. The two-stage detector that will be used in this master thesis is called Faster Region-Based Convolutional Neural Network (R-CNN). It will be discussed in more detail in Section 4.1.1.

2.1.3 One-stage Detector

In contrast to a two-stage detector, a one-stage detector directly predicts where the object could be in the image without the need for an RPN.

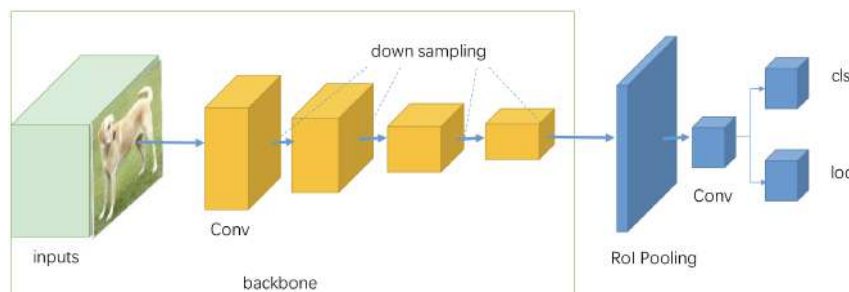


Figure 2.2: Basic architecture of a one-stage detector [Jiao et al., 2019]

Figure 2.2 shows a one-stage detector with a backbone network. The one-stage detector uses a fixed set of regions (anchors) instead of an RPN. The RoI Pooling layer uses the features from the backbone

network that are inside each of these anchors to detect and classify the object and the bounding box around the object [Jiao et al., 2019].

One-stage detectors are usually fast and good for real-time devices. The one-stage detector used in this master thesis is called You Only Look Once (YOLO). It will be explained in more detail in Section 4.1.2.

2.2 Dataset split

To train an object detector it is important to split the data into a training and a validation set. The training set is used to train the algorithm. To prevent the algorithm from overfitting on the given dataset, a validation set is used to test the algorithm's performance after training. The best model parameters are chosen based on the performance of the model on the validation set. Additionally, a certain amount of data should be used as a test set. After the model training and validation, the test set is used to get an idea of the general performance of the model [Xu and Goodacre, 2018].

2.3 Evaluation

The performance of an algorithm is evaluated based on the detection results on the test set, not the training set or validation set. The four basic terms used to evaluate the performance of a detection algorithm are [Minaee et al., 2022]:

- Ground Truth (GT): accurately labeled objects (annotations) in an image
- True Positive (TP): an object that is detected correctly
- False Positive (FP): the algorithm detected an object that does not exist
- False Negative (FN): an object (GT) was not detected

Based on these parameters it is possible to calculate Precision (P) (how many positive identifications are correct identification) and Recall (R) (how many GT objects were identified).

$$P = \frac{TP}{TP + FP} \quad (2.1)$$

$$R = \frac{TP}{TP + FN} \quad (2.2)$$

2.3.1 Log Average Miss-Rate

The Log Average Miss-Rate (LAMR) is a common value to determine the performance of a human detector. It is based on the Miss Rate (MR) (percentage of FN) and False Positives Per Image (FPPI)

for a definite number of images N . Nine FPPI reference points in the range of $[10^{-2}, 10^0]$ are used to calculate the LAMR with the following equations [Zhang et al., 2020]:

$$MR = \frac{FN}{TP + FN} \quad (2.3)$$

$$FPPI = \frac{FP}{N} \quad (2.4)$$

$$LAMR = \exp\left(\frac{1}{9} \sum_f \log(MR(\arg\max FPPI))\right) \quad (2.5)$$

The smaller the LAMR the better the model performed.

2.3.2 Intersection Over Union

In CV especially when using bounding boxes to define an object in an image, it is important to define the Intersection Over Union (IoU) metric. IoU defines the ratio between the overlapping area of two bounding boxes and the area of their union (see Figure 2.3). The two bounding boxes are the box around the GT object and the box around the Predicted Object (PO).

$$IoU = \frac{area(GT \cap PO)}{area(GT \cup PO)} \quad (2.6)$$

The IoU score of detection is between 0 and 1, where 0 means no overlap and 1 means perfect overlap [Koech, 2020]. Typically, scores from 0.5 up to 1 are counted as TP. However, selecting this threshold is up to the designer.

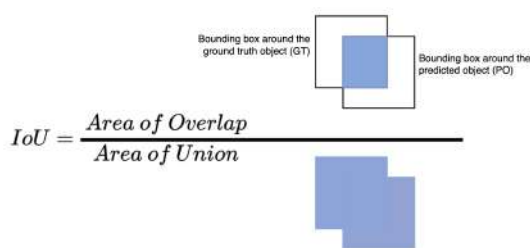


Figure 2.3: IoU formula; adapted of [Koech, 2020]

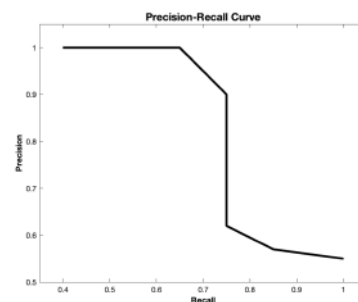


Figure 2.4: Precision-recall curve example; adapted of [Minaee et al., 2022]

Figure 2.4 shows an example of the precision-recall curve which states how the trade-off between P and R changes when changing the IoU threshold. Based on this curve, the user can select the right threshold that fits to the given problem.

In the case of detecting humans in water, it is better to have false positives than to not detect a person. Consequently, the threshold in this thesis is set to improve recall and reduce precision.

2.3.3 Average Precision

Nowadays a common value to evaluate the performance of a system is the Average Precision (AP) value that describes the area under the precision-recall curve. It offers a single value to describe the average of all predictions. The value AP is used, for example, by [Varga et al., 2022] and [Cafarelli et al., 2022] to evaluate different algorithms on their newly created datasets. The AP value at a certain IoU threshold α is defined by:

$$AP\alpha = \int_0^1 P(R)dR \quad (2.7)$$

Common metrics are AP50 (IoU = 0.5) and AP75 (IoU = 0.75).

Since AP is determined for each class individually, there are usually several values for the same dataset. To get a single value, the mean Average Precision is introduced. The mean Average Precision (mAP) defines the average of all AP values determined for all classes. [Koech, 2020]

$$mAP\alpha = \frac{1}{n} \sum_{i=1}^n AP_i \text{ for } n \text{ classes} \quad (2.8)$$

2.4 Fusion of detections

In addition to the selection and training of an individual model, the choice of fusion method for combining detections is important to achieve good results. Many successful classification methods combine results to improve the model's performance [Trick and Rothkopf, 2022].

Bayesian models are used to compute the likelihood of an observation and combine it with the Bayes' rule (probability of an event happening, depending on prior knowledge of conditions associated with the event). It is assumed that the detections are independently of each other. In a simple case considering only two images A , B and the detection results DR to be true or false, the OR rule, denoted \vee gives:

\vee	A_{true}	A_{false}
B_{true}	DR_{true}	DR_{true}
B_{false}	DR_{true}	DR_{false}

Table 2.1: OR rule

In this case DR is only false if both A and B are false. Thus the probability of DR being false is:

$$PropDR_{false} = PropA_{false} \times PropB_{false} \quad (2.9)$$

Assuming DR_{false} is a FN detection, the probability of DR being a TP ($PropDR_{TP}$), is simply:

$$PropDR_{TP} = 1 - PropDR_{FN} \quad (2.10)$$

[Schubert et al., 2004].

The concept given in Table 2.1 can be extended to a set of several observations. Each image adding another dimension to the table. In these more complex cases a detection result might be negative even through a certain amount of images show correct detections. For three images A, B, C considering DR to be a TP if at least two images have a TP, $PropDR_{TP}$ is:

$$\begin{aligned} PropDR_{TP} = & PropA_{TP} \times PropB_{TP} \times PropC_{TP} \\ & + PropA_{TP} \times PropB_{TP} \times PropC_{FN} \\ & + PropA_{TP} \times PropB_{FN} \times PropC_{TP} \\ & + PropA_{FN} \times PropB_{TP} \times PropC_{TP} \end{aligned} \quad (2.11)$$

If the probability for $A, B,$ and C to be a TP is the same, (2.11) can be pooled together:

$$PropDR_{TP} = PropA_{TP}^3 + 3 \times PropA_{FN} \times PropA_{TP}^2 = \sum_{i=2}^3 PropA_{TP}^i \times PropA_{FN}^{3-i} \times \frac{3!}{(3-i)! \times i!} \quad (2.12)$$

For indefinite number of images NI , with a minimum of N TP images, $PropDR_{TP}$ is:

$$PropDR_{TP} = \sum_{i=N}^{NI} PropA_{TP}^i \times PropA_{FN}^{NI-i} \times \frac{NI!}{(NI-i)! \times i!} \quad (2.13)$$

Using (2.13) any number of detection results can be fused to determine the likelihood of a certain observation. Since $PropA_{TP}$ and $PropA_{FN}$ function as a weight, this type of fusion is called Linear Opinion Pool [Berger, 2013].

3

State-of-the-Art

Contents

3.1	Detecting people on land overview	16
3.2	Existing water-based datasets	18
3.3	Variance in existing datasets	21
3.4	Literature Gaps	22

Lots of research in the field of human detection has already been done. Table 3.1 gives an overview of the research papers related to this thesis. The table is divided into two-stage and one-stage detectors to focus on comparing similar approaches while recognizing how different data/sensors collect that data, and influence the resulting model.

Two-stage detectors	Algorithm	Data	Sensor	Results
[Varga et al., 2022]	Faster R-CNN	Swimmers in open water	RGB cameras, including near to Infrared (IR)	54.7 AP50
[Cafarelli et al., 2022]	Faster R-CNN	Swimmers in open water	RGB camera	12.4 AP50
[Zhang et al., 2020]	Faster R-CNN	Different amounts of pedestrians	RGB camera	12.8 MR
[Nguyen et al., 2017]	CNN	Single pedestrians	RGB + Thermal camera	6 % EER
[Li, 2021]	Faster R-CNN	Pedestrians in different light settings	IR camera	26.2% mAP
One-stage detectors	Algorithm	Data	Sensor	Results
[Varga et al., 2022]	CenterNet	Swimmers in open water	RGB cameras, including near to IR	22.2 AP50
[Cafarelli et al., 2022]	YOLOX	Swimmers in open water	RGB cameras	12.6 AP50
[Lygouras et al., 2019]	YOLOv3	Swimmers in open water	RGB Camera	67% mAP
[Jacquelin et al., 2022]	tiny-Unet model	Videos of Swimmers in Pool	RGB camera	45 mAP 50
[Zhang et al., 2020]	RetinaNet	People on land in different amounts	RGB camera	31.47 LAMR
[Li, 2021]	YOLOv3	Pedestrians in different light settings	IR camera	34.9% mAP

Table 3.1: Overview Research Papers

Part of this master thesis is the creation of a new dataset that features swimmers from the point of view of an autonomous ferry. Table 3.2 provides an overview of existing datasets that feature swimmers, are taken from a boat perspective or are commonly used in human detection. To be aware of how others are creating their datasets, the number of images, the resolution of the images and the annotations are compared. Since this work relies on the ability to use existing datasets, it is important to provide information if a dataset is open source or not.

Swimmers	Dataset Name	Amount of Images	Resolution	Data Open Source	Single/Multi-Shot	Annotation
[Varga et al., 2022]	SeaDronesSee	54000	3840 x 2160	√	both	manually
[Cafarelli et al., 2022]	MOBDrones	126170	1920 x 1012	√	multi-shot	manually
[Leira et al., 2015]	<i>not named</i>	3000	720 x 480	x	multi-shot	manually
[Lygouras et al., 2019]	Swimmer Dataset	4500	416 x 416	√	single-shot	automatically
[Jacquelin et al., 2022]	Swimm ⁴⁰⁰	403	256 x 256	x	multi-shot	manually
Pedestrians	Dataset Name	Amount of Images	Resolution	Data Open Source	Single/Multi-Shot	Annotation
[Zhang et al., 2020]	WiderPerson	13382	1400 x 800	√	single-shot	manually
[Nguyen et al., 2017]	DBPerson-Recog-DB1	8240	204 x 20	√	single-shot	manually

Table 3.2: Overview Datasets

3.1 Detecting people on land overview

There are lots of different positions a human body can take. Based on that, there are also several different areas of detecting humans. Since there is more work already carried out detecting humans on land, an overview of this work is given first.

3.1.1 Pedestrian detection RGB camera

To detect pedestrians [Zhang et al., 2020] used an improved Faster R-CNN [Ren et al., 2015] and a RetinaNet (one-stage detector) [Lin et al., 2017] architecture to detect people and compare their created dataset (examples in Figure 3.1) with other existing datasets. To evaluate the model performance the authors used the LAMR. They obtained a 12.8 LAMR for Faster R-CNN, while only 31.47 LAMR for RetinaNet.



Figure 3.1: Example detection results [Zhang et al., 2020]

3.1.2 Pedestrian detection using thermal and IR cameras

The goal of [Nguyen et al., 2017] is to reduce noise in pedestrian detection by combining RGB and thermal cameras. They achieved an Equal Error Rate (EER) below 6% following this process: Both images were captured at the same time by a dual camera. The algorithm was trained separately with each type of image. In the end, the extracted features were combined to detect the person correctly (Figure 3.2).

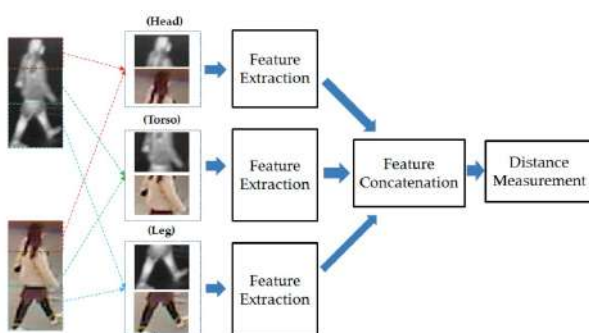


Figure 3.2: Combining RGB and Thermal Images



Figure 3.3: Usage of IR cameras

To decrease the influence of high-heat objects close to human bodies, [Li, 2021] suggests to use IR cameras instead of thermal cameras. The authors obtained better results using YOLOv3 [Redmon and Farhadi, 2018] (mAP 34.9) to detect pedestrians in IR images (Figure 3.3) than using Faster R-CNN [Ren et al., 2015] (mAP 26.2).

3.2 Existing water-based datasets

There are several big datasets available when it comes to detecting people in land-based scenarios. Among the biggest are [INRIA](#), [Caltech](#) and [KITTI](#). On contrary, the amount of datasets including people in water is very small.

3.2.1 Detecting people in water from Unmanned Aerial Vehicle (UAV)

The SeaDroneSee dataset [[Varga et al., 2022](#)] aims to fill this gap by creating a UAV dataset for Search and Rescue (SAR) missions. The dataset includes 54,000 images (examples in Figure 3.4) divided into swimmer, floater (swimmer with a life jacket), life jacket, swimmer on a boat, floater on a boat, regions to ignore (like land) and unlabeled objects (for example, wood). Pictures were taken with several cameras (including an IR camera) to minimize the effect of camera bias (all images taken from the same viewing angle). They obtained the best result (54.7 for AP_{50}) using a Faster R-CNN [[Ren et al., 2015](#)] while only 22.2 for AP_{50} when using the one-stage detector CenterNet [[Zhou et al., 2019](#)].

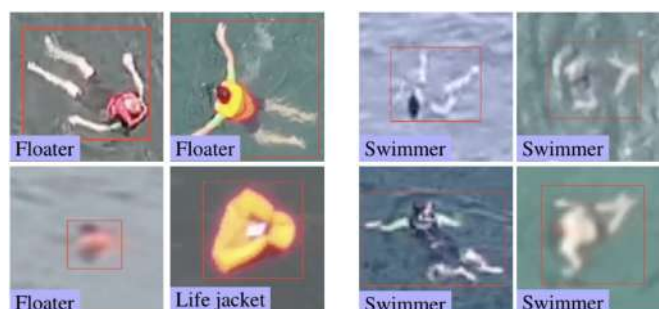


Figure 3.4: Categories in [[Varga et al., 2022](#)]

Another big UAV based dataset is the MOBDrones [[Cafarelli et al., 2022](#)] dataset with 125,000 drone-view images (examples Figure 3.5) of which 72% feature humans in overboard situations. To annotate all these images, the open-source software Computer Vision Annotation Tool (CVAT) [[opencv, 2022](#)] was used. Based on these annotations the authors were able to obtain a 37.8 AP_{50} with a two-stage detector (12.4 AP_{50} using Faster R-CNN [[Ren et al., 2015](#)]), and 12.6 AP_{50} with YOLOX [[Ge et al., 2021](#)].



Figure 3.5: Zoomed in detection results [[Cafarelli et al., 2022](#)]

SAR missions using UAVs might also happen at night or during weather conditions with poor visibility. Therefore, [Leira et al., 2015] used IR cameras to detect people in open water. While using a simple edge detector, 98.5% of humans were detected correctly and 93.3% were classified correctly. However, the model did not perform well whenever the horizon was in the image due to the simplicity of the used edge detector. This shows that within certain limits, using thermal images in an ocean scenario gives good detection results.

As [Leira et al., 2015] does not offer its dataset publicly, the only available IR images from the dataset is given in Figure 3.6.

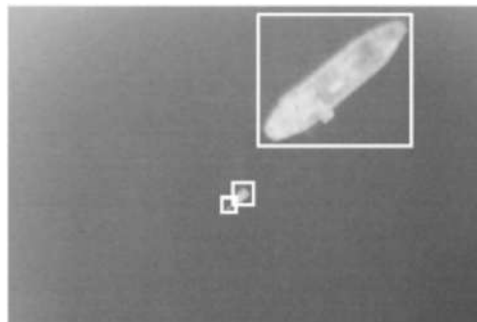


Figure 3.6: Example edge detector [Leira et al., 2015]

3.2.2 Working with small datasets

In [Lygouras et al., 2019], it is presented the Swimmer Dataset containing images with people swimming, captured from various angles and altitudes (example images Figure 3.7). Since the dataset only contains 4500 images, the author used the negative of each image as well. Additionally, they retrained their algorithm with all the images containing FP detections to improve their network. With this approach, they obtained 67% *mAP* and 70% recall on a YOLOv3 [Redmon and Farhadi, 2018] network. While using MobileNetV2 [Sandler et al., 2018] they only achieved a *mAP* of 21%.



Figure 3.7: Examples Swimmer Dataset

The *Swimm*⁴⁰⁰ dataset created by [Jacquelin et al., 2022] is a very small dataset with only 403 images. They were still able to obtain a 45% mAP50 by using the following augmentation strategies to increase the model's performance:

- Zoom-in: cropping the image to make the swimmer be bigger
- Zoom-out: adding neutral colour around the swimmer to make the swimmer look smaller
- Side-switch: moves swimmer from the center to the side of the image
- random left-right flip: to double the dataset
- colour change: thus the water had all different types of blue and even green
- contrast and brightness variations: to simulate different weather conditions
- Gaussian blur: for overall robustness

Figure 3.8 shows examples of seven different augmentations used in [Jacquelin et al., 2022]. From left to right, top to bottom: original image, blur, contrast and brightness change, crop, horizontal flip, hue change, side switch, zoom out.

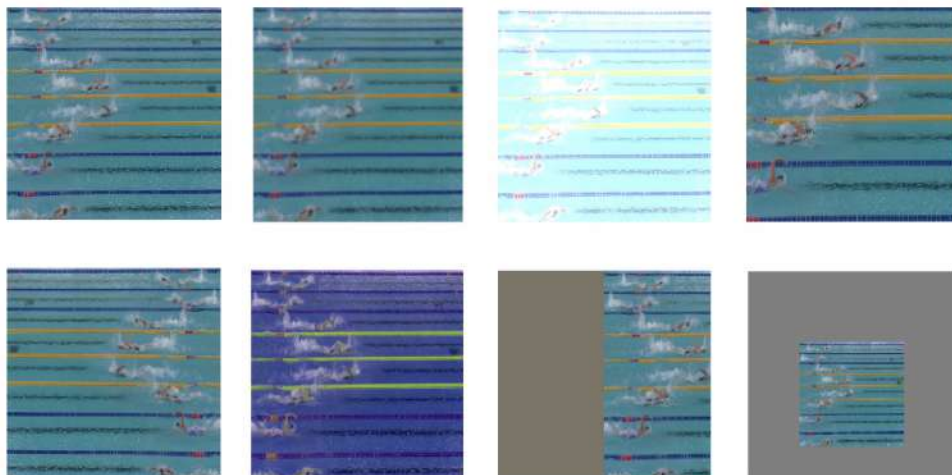


Figure 3.8: Augmentation examples [Jacquelin et al., 2022]

3.3 Variance in existing datasets

To gain a better understanding of the available data, the variance in the existing two datasets - SeaDronesSee [Varga et al., 2022] and Swimmer Dataset [Lygouras et al., 2019] - was analyzed. Table 3.3 and Table 3.4 show the number of images per defined category. The categories for the two datasets are unrelated because the two datasets focus on very different situations. Most images in the SeaDronesSee dataset include more than two people and show them in the same size or smaller than the people in the "Far away" category of the Swimmer Dataset. In contrary the Swimmer Dataset features swimmers in very different sizes with the biggest ones almost filling the full image.

Table 3.3 shows that the amount of swimmers in the SeaDronesSee dataset is minimal. However, they do have a lot of floaters. A floater is a swimmer with a life jacket on. For the ferry, it does not matter if the person has a life jacket (it is not in an SAR mission). Thus the floaters and swimmers fall in the same category for further training of the models (see Appendix A.1.1 for verification).

	Training set	Test set	Validation set
Only swimmers	5	15	7
Only floaters	207	184	96
Swimmers & others	349	24	147
Images with black squares	87	81	62
Images with a very small swimmer	1669	1156	409

Table 3.3: Amount of images per category in the SeaDronesSee dataset

As Table 3.4 shows, the amount of black people and children in the Swimmer Dataset is small (SeaDronesSee does not have any nonwhite people or children). The same accounts for the situation of sunrise/sunset, a partially occluded person, a person reflected by the water surface and a swimmer with their head underwater.

	Close to the camera	Medium distance	Far away
1 person	234	394	224
2 people	122	129	408
3 or more people	198	293	451
people of color	12	1 (+10 crowd scene)	1
kids	36	0	0
Partially occluded person	0	5	297
Sunrise/Sunset	6	0	0
Swimmer reflected by water surface	4	1	0
No head shown	0	14	0
Swimmer + Animal	4	120	161
Swimmer + Other objects	18	353	60
people in a swimming pool	336	404	0

Table 3.4: Amount of images per category in the Swimmer Dataset

3.4 Literature Gaps

In this chapter, the relevant state of the art was reviewed. The most recent work done on swimmer detection is either limited to swimming pools or based on the view of a UAV during an SAR mission. A detection model meant to detect swimmers in open water from the perspective of a small vessel is not yet created. This is the gap we aim to address in this work.

4

Methodology

Contents

4.1 Algorithms	24
4.2 Swimmer detection model	26

The goal of this master thesis is to create a model to detect swimmers in open water. The developed approach starts with collecting the data from the viewpoint of a ferry. Then two different CV algorithms will be trained with that collected data as well as with already existing datasets of swimmers. Finally, the performance of each algorithm will be evaluated.

4.1 Algorithms

As explained in Section 2.1 different algorithms can be used for object detection. Similar to [Varga et al., 2022] and [Zhang et al., 2020] the performance of the dataset will be evaluated on both a two-stage and a one-stage detector. Based on the current state of the art, the two-stage detector is Faster R-CNN and the one-stage detector YOLO.

4.1.1 Faster R-CNN

The network Faster R-CNN evolved from R-CNN. It is a two-stage detector. Consequently, the network first proposes a region where an object could be and then in the next step, classifies what object is there. While R-CNN relied on a selective search algorithm to create region proposals, [Ren et al., 2015] has replaced the selective search algorithm with an RPN to create Faster R-CNN. RPN is a neural network that determines anchor boxes. It is able to create several anchor boxes simultaneously, as shown in Figure 4.1. This improved the network's efficiency.

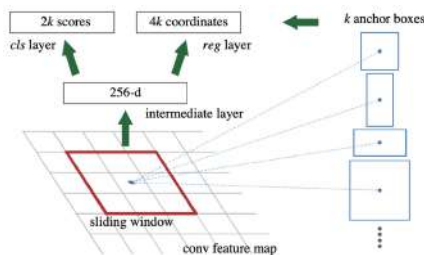


Figure 4.1: Region Proposal Network (RPN) [Ren et al., 2015]

The RPN uses the output of the backbone network as the RPN's input. A sliding window moves over the feature map given by the backbone network. Based on the output of the sliding window, the box-classification layer (*cls* layer) predicts the probability of objects being in the image. Simultaneously, the box-regression layer (*reg* layer) determines where in the image the potential object is and what size it has. Thus, the RPN outputs anchors of three scales and three ratios which are further used in the Faster R-CNN network to classify the object and create the bounding box around it. Faster R-CNN is built to compute faster than a traditional R-CNN. At the same time, it is precise and efficient. [Jiao et al., 2019] For this work Faster R-CNN with a Resnet50 backbone was used.

4.1.2 YOLO

The YOLO network was first developed by [Redmon et al., 2016]. It is a one-stage detector designed for real-time object detection. The architecture of the network is shown in Figure 4.2.

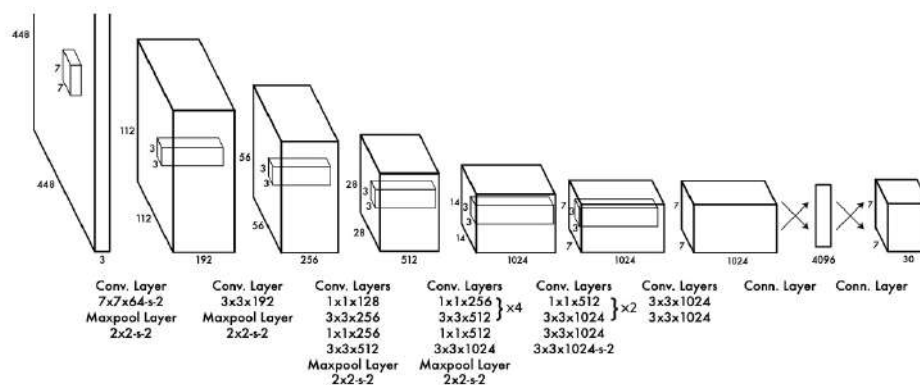


Figure 4.2: Region Proposal Network (RPN) [Ren et al., 2015]

YOLO has 24 convolution layers followed by two fully connected layers. The network divides the image into grid cells and predicts the bounding boxes in the pixel that contain an object as well as the probability of the cell belonging to a certain class. The network is trained on a loss function that corresponds to how well it performs the detection task. It is a mix of cross-entropy (how likely the object belongs to a certain class) and regression loss (if the box has the right size). The loss function is the same for big and small objects. It focuses on bounding box predictions rather than confidence predictions. Classification errors are only penalized if there is an object present in that grid cell.

After the first YOLO algorithm was released, there were several improvements. YOLOv2 and YOLOv3 were improvements made by the creator of YOLO by changing the backbone of the network. Both times the algorithm became faster than it was before. The newest version is YOLOv8 created by ultralytics. YOLOv8 is an anchor-free model with a decoupled head (Figure 4.3) similar to YOLOv7 that independently processes object detection, classification and regression tasks due to its decoupled head. It uses a modified CSPDarknet53 [Wu et al., 2020] backbone related to YOLOv5. Based on this design the authors were able to improve the model's accuracy compared to other YOLO models. [Terven and Cordova-Esparza, 2023]

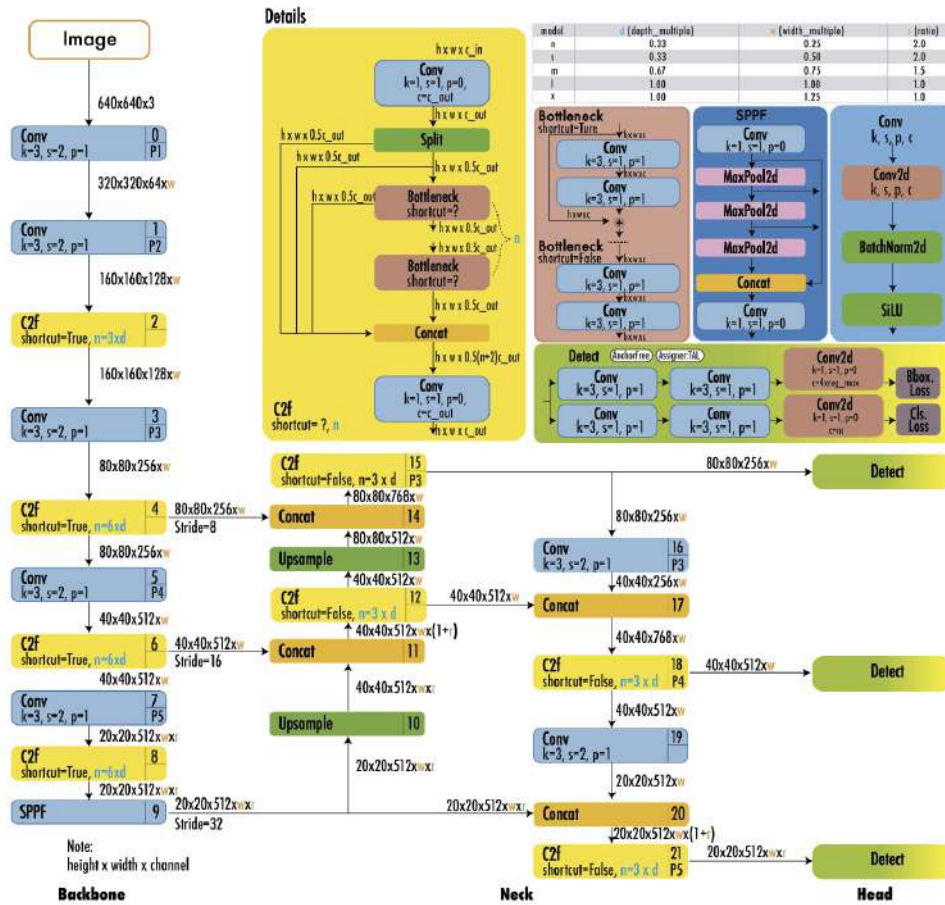


Figure 4.3: YOLOv8 architecture [Terven and Cordova-Esparza, 2023]

4.2 Swimmer detection model

To set up a good model, several steps are needed after creating a dataset and deciding on the algorithm to use. Due to the size of the dataset the data will first be augmented. Then, the different algorithms will be trained and finally evaluated. This process will lead to a good swimmer detection model.

4.2.1 Data augmentation

Since the dataset, as described in Chapter 6, will be created in several sessions, there will be some variation in the data. However, it is important to acknowledge that it won't be possible to cover all imaginable situations. Thus, the variation of the training data will be increased by using the augmentation methods suggested in [Jacquelin et al., 2022]. That means that for the RGB images different weather conditions will be simulated by changing the contrast and brightness of the images, as well as adjusting the colour to create different shades of blue for the water. Additionally, for both image types, swimmers

will be positioned in different spots of the image by using side-switch, random left-right flip and zoom-in/zoom-out. Lastly, Gaussian blur will be used to increase the overall robustness of the model.

4.2.2 Training

The training of the two algorithms introduced in Section 4.1 (Faster R-CNN and YOLOv8) is based on the benchmark SeaDronesSee data set from [Varga et al., 2022] which features images of swimmers taken by RGB cameras and near to IR cameras. They only obtained results for Faster R-CNN which will be verified with their available dataset. In reference to this Faster R-CNN and YOLOv8 will be trained on the new dataset plus some images from the Swimmer Dataset [Lygouras et al., 2019] which come closer to the camera position of the ferry.

4.2.2.A Transfer Learning

Training a model from scratch takes a lot of time and resources. However, it is possible to reuse a CNN on related problems. Transfer learning means that the weights of the pre-trained model are reused. Consequently, researchers often use transfer learning ([Lygouras et al., 2019], [Zhang et al., 2020], [Yassine et al., 2020]) to retrain an already well-trained model on new limited data.

The data available in this thesis is also limited. Therefore, algorithms that are already trained on detecting humans will be retrained with the newly created dataset. Both Faster R-CNN and YOLOv8 pre-trained on MS COCO are available [PyTorch, 2017], [Ultralytics, 2023] and will be used for this master thesis.

4.2.2.B Cross-validation

During the training of the algorithm, it is important to check the model's robustness across different parts of the dataset. To make sure that the dataset division presented in Section 6.3.1 is not biased, cross-validation will be used.

Cross-validation means that the dataset is randomly divided into k mutually exclusive subsets, in this case, $k = 5$. Then the training, validation and testing of the model are done k -times. Based on this the variance of the predictions can be checked. If the predictions show only a small variance, it means the model performs consistently in various dataset splits and is assumed to achieve good results with similar new data as well. [Kohavi et al., 1995]

4.2.3 Evaluation

Based on the problem given in Section 1.2 two different parameters are important for a good-performing model: AP (to make sure every swimmer is detected) and speed (aim to work in real-time).

4.2.3.A Object detection performance

For an autonomous ferry, it is very important to detect every possible obstacle on its path while not misclassifying open water as an obstacle. As a missed swimmer means the ferry might harm a person, the model should maximize its recall rate (detecting as many swimmers as possible) to secure a safe journey. As described in Section 2.3.3 the performance of the model can be evaluated using the AP value. Since the results are compared to the baseline [Varga et al., 2022], the AP_{50} will be used to compute the detector's performance.

4.2.3.B Detection Speed

When navigating anywhere the captain has to make decisions very fast. Consequently, the machine also has to be very fast to detect objects to give time for the rest of the system to decide how to operate the ferry. Proceeding from there, the different algorithms will be compared with regard to the time each algorithm needs to process a certain amount of information. To compare the speed, all algorithms will be trained with the same images and afterwards tested with the same test set on the same device to reduce biases based on different equipment.

5

Preliminary Experiments

Contents

5.1 Influence of distance between camera and swimmer	30
5.2 Influence weather	31
5.3 Influence variety of swimmers	33
5.4 Influence amount of people in the image	36

Prior to creating a new dataset, several tests with the existing two datasets - SeaDronesSee [Varga et al., 2022] and Swimmer Dataset [Lygouras et al., 2019] - have been done to understand which images exactly are needed. With the goal in mind to assure that a swimmer in every possible situation in front of the ferry will be detected, a Faster R-CNN model was trained and tested on various situations which will be explained further in this chapter. For these tests Faster R-CNN was chosen because it offers to randomly assign images to the training and validation set which decreases the volatility in the training of the algorithm.

5.1 Influence of distance between camera and swimmer

Section 3.3 showed that both datasets have images featuring swimmers close to the camera and far away. To get a better understanding of the influence the size of a swimmer in the image has on the detection results, experiments with different setups were done.

When training the algorithm only with people far away from the camera (using images from both datasets), people close by might not be detected at all as shown in Figure 5.1. Adding images featuring swimmers at a medium distance to the training set increases the performance of the model. However, the same swimmer might still not be detected correctly as shown in Figure 5.2.



Figure 5.1: Swimmer not detected

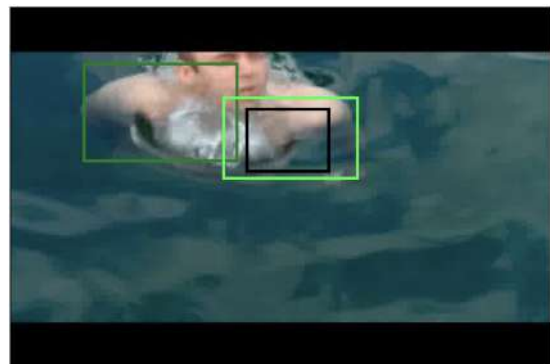


Figure 5.2: Swimmer detected as several swimmers

In the opposite way training the algorithm only with images featuring swimmers close to the camera - again containing images from both datasets - will affect the results on swimmers far away from the camera. As Figure 5.3 shows only the rescue can was detected twice but no swimmer. After adding the images featuring swimmers in a medium distance, at least one of the swimmers was detected as well (Figure 5.4).



Figure 5.3: Swimmers not detected



Figure 5.4: One swimmer detected

Based on these findings, it is important that the newly created dataset features swimmers at different distances from the camera.

5.2 Influence weather

The ferry will run no matter the weather. Hence, it might be rainy, it might be sunny, and it might be cloudy. Additionally, the sea might be rough or very calm. If the sea is rough, people might be occluded by waves and swell.

The Swimmer Dataset includes 297 images with people far away from the camera that are partially occluded by waves. As SeaDronesSee has no occluded people in its images (all taken during a calm day), images from this dataset were not included in this test. Without having any pictures with occluded people in the training set, the model was still able to detect all swimmers in 249 images. In 19 images the model had at least one false negative.

Figure 5.5 to 5.8 show the three situations when the model did not detect all the swimmers. In Figure 5.6 it would not be a problem because the two humans are close together. In contrast to Figure 5.7 and 5.8 the ferry would hit the not detected swimmer by trying to avoid the detected ones. This will cause dangerous situations.



Figure 5.5: Swimmer occluded



Figure 5.6: One swimmer detected



Figure 5.7: One swimmer detected

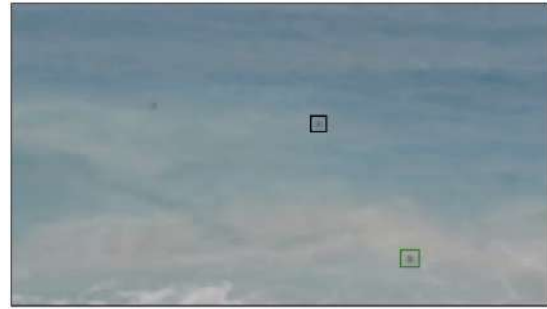


Figure 5.8: Two out of three detected

In opposite to a very rough sea, on a very calm day, the water's surface reflects the shore but also everything on the water. As the Swimmer Dataset and SeaDronesSee both feature images of people swimming during a calm day, they were mixed to have people of both datasets present in the training and in the test set.

In the Swimmer Dataset are five swimmers who are reflected on the surface while swimming. All of them were detected correctly. Two detections included the swimmer and their reflection (example in Figure 5.9) and one image had the reflection detected as a separate swimmer (Figure 5.10). The increment in size as shown in Figure 5.9 makes the algorithm believe the swimmer is closer to the camera than it actually is. Even though this will cause the ferry to stop earlier, it is not considered a problem since the ferry stopping earlier does not harm the swimmer.



Figure 5.9: Reflection as part of swimmer

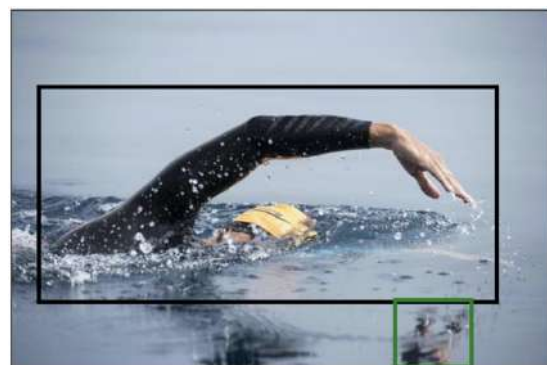


Figure 5.10: Reflection as separate swimmer

Besides swimmers being reflected while swimming, humans sitting on a boat (images from SeaDronesSee) can also be detected as swimming in the water (Figure 5.11 and 5.12). Since it is a reflection, the "swimmer" will always stay close to the boat. This false positive will therefore not influence the performance of the ferry.



Figure 5.11: Reflection of person on boat



Figure 5.12: Reflection of person on boat

In the morning or in the afternoon when the sun is low in the sky it will get reflected by the water. As shown in Figure 1.5 the reflected sunlight might over-saturate the camera sensor making it difficult to recognize swimmers. As all the images in the SeaDronesSee dataset were taken during the middle of the day, their images were neither included in the training set nor in the test set. There are only six images which are taken during sunrise/sunset in the Swimmer Dataset. However, the model can only detect half of them correctly (example Figure 5.13). In the other three images, the reflected sun is part of the detections (example Figure 5.14).



Figure 5.13: Correct detection

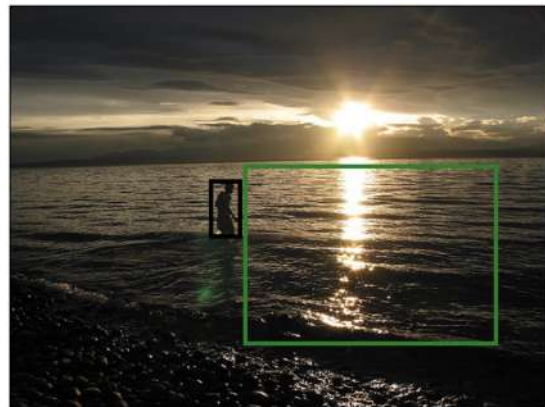


Figure 5.14: Reflection of the sun detected as a swimmer

5.3 Influence variety of swimmers

An algorithm needs to be shown images of various people to assure it does not specialize on one person. Even with a variety of people in the image, they might still all belong to the same social group or just do similar gestures.

All images in SeaDronesSee show white swimmers while the Swimmer Dataset has 12 images with people of color. As there is an image sequence in the SeaDronesSee which matches closely to the images of people of color, these images form the training set, together with images of white people from the Swimmer Dataset. The results for black swimmers were good (example Figure 5.16) while the

algorithm has trouble detecting humans with brown skin (example Figure 5.15). While the amount of images with people of colour is very small there are a lot of white swimmers wearing black wetsuits, especially in the images from SeaDronesSee. Thus, the model is used to see body parts (example arms) in different tones. Based on these detection results, it is helpful to have some swimmers wear wetsuits when recording images.



Figure 5.15: Swimmer with brown skin



Figure 5.16: Swimmer with black skin

Another group of swimmers, who are underrepresented in the datasets, is children. SeaDronesSee only features students, while the Swimmer Dataset has people of all ages. There are 36 images with only children on them and 8 with a child and an adult. As there are no children in SeaDronesSee, the model was only trained with images from the Swimmer Dataset. The test set had all images with children on it, independently if they depicted adults as well or not. Most of the children were detected correctly. An interesting fact is: None-white children were all detected accurately in this trial (example Figure 5.17). In the test regarding the influence of human skin colour, they were not detected correctly. (example Figure 5.15). This suggests that training the model with people with different skin colours is more important than training with people from different age groups.

All images that missed a detected child were group images with the group being detected as a person, not the individuals. This is not a problem for the situation considered in this thesis. First, it will not be likely for the ferry to encounter a group of children as shown in Figure 5.18. Second, even if exactly this situation will happen, it is good when the ferry stops immediately upon detecting this person very close to the camera to assure that the children are safe.

As there was no image showing children where the children were not detected at all, we can expect the model to detect children with the same probability as adults even if it is only trained on adults.

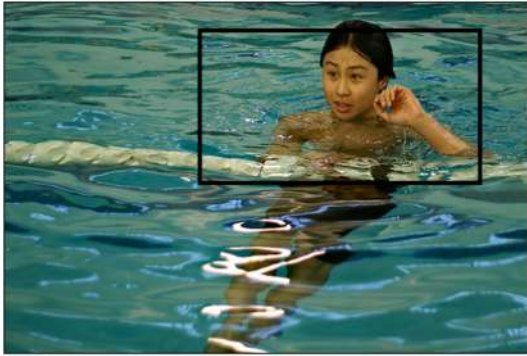


Figure 5.17: Single child



Figure 5.18: Group of kids

Typically, when a person is swimming, their head will be above the water or very close to the water's surface. This could cause the model to detect a swimmer only based on their head. In case someone is drowning, their head might not be above the water but it is very important to detect this person. Thus, images of humans without a visible head will form another group of swimmers. For the recordings of SeaDronesSee, swimmers were offered life jackets and are mostly floating on the water's surface consequently there are no images without a visible head. To reduce the influence of different datasets on the detection results, only images from the Swimmer Dataset were used to train the Faster R-CNN model.

The model had a lot of trouble detecting people without a visible head. Out of 15 images only three were detected correctly. Often the human was just expected to be a lot bigger than the ground truth (see example Figure 5.19). When having several "not connected" body parts, they are detected but might not be detected as one person (example Figure 5.20). The expected size of the "two" humans (each represented by one arm) is at maximum half the size of the real human in the image. This is a problem because the ferry would expect the swimmer to be further away than they actually are.

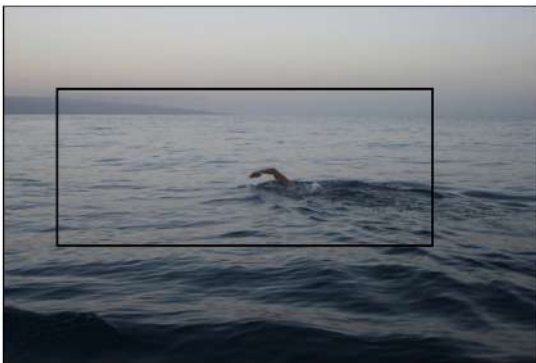


Figure 5.19: Person detected too big

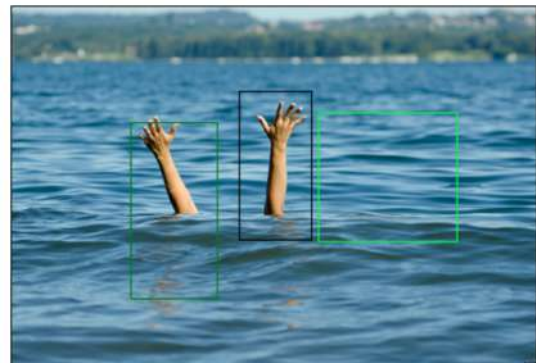


Figure 5.20: Only arms

5.4 Influence amount of people in the image

For this work, we expect to have a maximum of two swimmers in front of the ferry. In both datasets SeaDronesSee and Swimmer Dataset there are a lot of images showing more than one person. When using these datasets the algorithm will consequently get used to having several positives in the image. The goal of this test was to determine if the amount of FN and FP changes depending on the amount of swimmers on the images the model is trained with. For this test two sets of images were created featuring images from both datasets: One with images with a maximum of one swimmer in the image and the other set with images with more than two swimmers in the image. To obtain results in both directions, each set was once used as the training set and once as the test set.

Training the algorithm only with images with more than two swimmers leads to more false positives compared to training with only one swimmer and testing on images with several swimmers. Table 5.1 indicates that training the algorithm exclusively on images containing a single swimmer will result in the algorithm not detecting all swimmers when there are multiple individuals present.

Test set	More than 2 people	One person
Correct detections	631	555
False positives	116	377
False negatives	598	88

Table 5.1: Influence amount of people

Consequently, it is important to have images with different amounts of swimmers in the dataset. This holds true even if the situation this work is based on will never have more than two swimmers in front of the camera.

Many images with several swimmers on them also show objects and/or animals. Since this work uses transfer learning and models, which are trained on MS COCO, it is likely that the algorithm will detect the animals and objects as well. This is good as the ferry should not ram any object or kill animals. Appendix A.1.2 and A.1.3 verify that anything in the image which is not a human does not affect the detection of swimmers even if these objects and/or animals were not in the training images.

SeaDronesSee also features images with black squares which are used to reduce the number of boats in the image. For this thesis the ratio between boats and swimmers is not important, thus we will not work with images with black squares.

The Swimmer Dataset includes 740 images taken of people in a swimming pool. Even though these images show swimmers, they will not be included in this work, after all a passenger ferry will never be found in a swimming pool. Additionally, the conditions of the water surrounding the swimmer are a lot different from an open-water scenario.

6

Dataset creation

Contents

6.1 Cameras	38
6.2 Annotation Method	39
6.3 New dataset	40

The swimmer dataset needed for this thesis will be acquired in Portugal due to the good weather conditions in spring. Since this work is based on a ferry, its travel conditions are taken into account when creating the dataset. Figure 6.2 shows the area where the ferry is operating. The ferry transports day tourists to an island. Thus the dataset acquisition will happen during the daytime. There will be three recordings at the beach (Figure 6.1) as well as one at a lake to include different weather and water situations.



Figure 6.1: Beach (image recording location)



Figure 6.2: Helsinki (ferry operation location)

6.1 Cameras

The equipment used to record data are a RGB camera and a thermal camera provided by Callboats, as well as the camera of an iPhone 12 mini. As shown in Figure 6.2 the water is calm and open space. Based on the dimensions of the ferry the cameras are mounted $2.5 - 3m$ above the water level.

6.1.1 RGB Camera

Most swimmer datasets use RGB cameras ([Varga et al., 2022], [Cafarelli et al., 2022], [Lygouras et al., 2019]). The advantage of an RGB image is that there is more information in the image that can be used for detection than in a grey-scale image.

The RGB camera that is used in this thesis is shown in Figure 6.3. It is a GV-TDR2700 camera from [geovision](#). The resolution of the images is 1280×720 and the camera is able to record 30 images per second. However, due to the segmentation task splitting the water area from the shore and the limited computational power on board the ferry, only one image per second can be processed. The camera has a defog option for better visibility in bad weather.



Figure 6.3: RGB Camera [Inc, 2021]



Figure 6.4: Thermal Camera [LLC, 2021]

6.1.2 Thermal Camera

In times with poor visibility, thermal cameras have the advantage of looking for heat radiation. Additionally, they reduce the effect of background, clothing and accessories when recognizing humans during the day. [Leira et al., 2015] proved that object detectors can score good results when using thermal images. The thermal camera used in this work is FLIR E4. Figure 6.4 shows an image of the camera. It has an IR resolution of 80 x 60 and a thermal accuracy of $\pm 2C$ for ambient temperature 10C to 35C. The camera can record 9 images per second. Nevertheless, each image has to be taken manually ergo it is not likely to record 9 images per second.

6.1.3 iPhone Camera

To decrease the influence of the used camera on the model, the camera of an iPhone 12 mini will be used as a third device to record images. It has a resolution of 2340 x 1080 for taking images and is able to record videos with 30 images per second at a resolution of 1920 x 1080. Thus, the iPhone will be used for both: Recording videos and single images. Combining these two creates a larger amount of images, while obtaining some images in a high resolution at the same time.

6.2 Annotation Method

After the images have been recorded, they have to be manually annotated to provide the ground truth for the CV algorithm. As suggested by [Cafarelli et al., 2022] the software CVAT [opencv, 2022] will be used to help annotate the images. CVAT is an open-source software which describes itself as an "industry-leading data engine for machine learning" [opencv, 2022]. The software was created by Intel to facilitate image and video annotation through interpolation and segmentation.

6.3 New dataset

Based on the results of Section 3.3 specific situations were selected to focus on with the new dataset. As Section 5.2 has shown the weather has a big influence on the performance of the detector. As a result, additional images were captured during sunset, as shown in Figure 6.5, as well as more images that contain people partially (Figure 6.6) and fully (Figure 6.7) occluded by waves.

6.3.1 Dataset size

As explained in Section 6.1, three different cameras were used to record the data. Table 6.1 shows how many images were recorded with which camera and what the resolution of the recorded images is.

Camera	Recorded Images	Image Resolution
RGB Camera	1119	1280 x 720
Thermal Camera	76	80 x 60
iPhone	206	2340 x 1080
	10 videos	1920 x 1080

Table 6.1: Recorded images per camera

To have a better variety in the dataset, there have been four individual recording sessions. Table 6.2 gives an overview of the distribution of how many images were recorded during each session as well as an average of the swimmers annotated per image.

Recording	Recorded data	Average Number of Annotations
Sunset 1	2 videos	9
Beach daylight	608 images	7,7
	2 videos	1,5
Sunset 2	793 images	9,6
	5 videos	3,8
Lake	1 video	1

Table 6.2: Recorded images per recording

Table 6.3 shows how many images of each category the new dataset contains. Some images could fit into several categories so they were placed in the category containing fewer images. For example, all images with the black person were taken during sunset (see Figure 6.9). However, they were still placed in the category "people of colour".

Type	Annotated Images
People of color	99
People without a visible head	191
Occluded people	51
Sunset	989
One person	358
Two people	49

Table 6.3: Overview images per category

6.3.2 Example Images

After having an overview of all the images taken during the recordings, this section presents examples of the images in the dataset. As presented in Table the images were divided in several categories. Figure 6.5 shows examples of the images recorded during sunset. The left two images were recorded with the iPhone during the first sunset recording while the two images on the right side were recorded during the second sunset with the camera from Callboats (upper right), as well as with the iPhone (lower right).



Figure 6.5: Examples of swimmers during sunset

Figure 6.6 shows images of partially occluded swimmers recorded with the camera from Callboats. Figure 6.7 shows on the bottom left an image of a swimmer underwater recorded with the iPhone and on the top right a swimmer under water recorded with the camera from Callboats. Most images with occluded people were taken during daytime. Only three images with partially occluded people were taken during sunset.



Figure 6.6: Examples of swimmers partially occluded



Figure 6.7: Examples of swimmers fully occluded

As presented in Section 5.3, the variety of swimmers especially if their head is visible or not had a big influence on the detection result. Thus, a commercial diver and an athlete swimmer ensured that it was possible to create images of only feet or arms being visible in the image through diving in the water and freestyle swimming. People jumping into the waves also created images of humans in the water without a visible head. Figure 6.8 shows examples of both scenarios. The two images on the left side were taken with the camera from Callboats, the two on the right were taken with the iPhone.



Figure 6.8: Examples of swimmers without a visible head

Section 5.3 also experimented with the influence of kids on detection results. As all the kids were detected even when only training with adults, kids were not included in this dataset.

To assure detection results are not based on race, a black swimmer participated on the recordings during sunset. The image on the left side in Figure 6.9 is was recorded with the camera from Callboats, the image on the right side with the iPhone.



Figure 6.9: Examples of a black swimmer during sunset

Some of the images taken during the recordings do not show any special situation or person. These images are placed in the groups "one person" (Figure 6.10) and "two people" (Figure 6.11) in Table 6.3. All images shown in Figure 6.10 were taken with the iPhone during daytime. The leftmost image shows the recording taken at the lake, the other two are from the beach during daytime. The images depicted in Figure 6.11 are all taken at the beach during daytime. The leftmost image was taken with the camera from Callboats, while the other ones were taken with the iPhone.



Figure 6.10: Examples of images with one person



Figure 6.11: Examples of images with two people

Lastly, there were also 25 images taken (Examples in Figure 6.12) with the thermal camera presented in Section 6.1.2. As the camera used IR to create the thermal images, it did not provide the results hoped for. For a human, it is a lot harder to detect a swimmer in these thermal image than in a normal images. The model showed the same results. Thus, only RGB images taken with this camera were included in the final dataset.

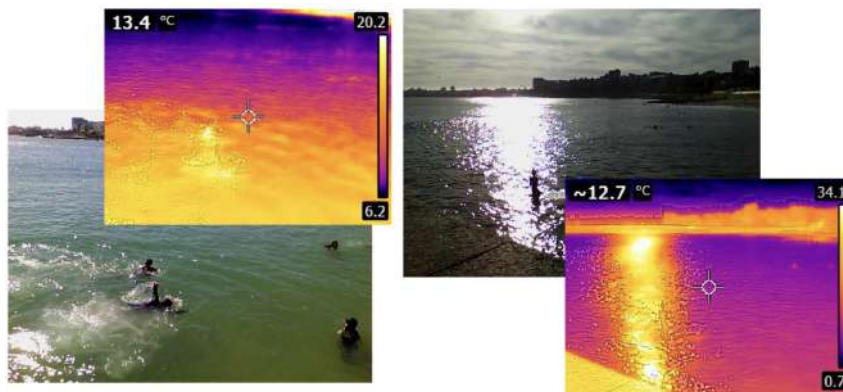


Figure 6.12: Images taken with a thermal camera

7

Results

Contents

7.1 Dataset preparation	46
7.2 Faster R-CNN	51
7.3 YOLO	53
7.4 Comparison Faster R-CNN and YOLOv8	57
7.5 Probability FN and FP per season	61
7.6 Implementation model	65

This thesis aims to create a model that detects swimmers in water, based on the view of a ferry. The new dataset presented in Section 6.3 was combined with some of the images from the Swimmer Dataset, which are taken from a similar perspective, to create a dataset with a big variety of images that all come close to how the ferry will see a swimmer.

Before Faster R-CNN and YOLO can be trained with this combined dataset, there was some preparation done:

- A cross-validation was done to check the variance of the dataset.
- The dataset was split into training, validation and test set.
- To improve the training results, the data was augmented which increased the variety.

Then Faster R-CNN and YOLOv8 have been trained and evaluated individually. After fine-tuning the results of each model they are compared to determine which model performed better on the given task.

7.1 Dataset preparation

As CV tasks highly depend on the data used and having more data implies covering more situations. The new dataset was combined with some images from the Swimmer Dataset that have a similar perspective as the viewing angle the camera on the ferry has.

The dataset used to obtain the results presented in this thesis includes a total of 3865 images. Approximately half of the images are from the new dataset and half are from the Swimmer Dataset. Table 7.1 gives an overview of how many images from each dataset belong to which category. By including images from both datasets, it is possible to have a well balanced dataset with more than 100 images for each distinct category.

Type	Images New Dataset	Images Swimmer Dataset	Total images
People of color	99	12	111
People without a visible head	191	15	206
Occluded people	51	297	348
Sunset	989	6	995
One person	358	613	971
Two people	49	578	627
More than two people	0	607	607

Table 7.1: Overview images used for final training

7.1.1 Cross-Validation dataset

As presented in Section 4.2.2.B the dataset was divided into 5 equal, randomly chosen sets for the cross-validation. After running the cross-validation with Faster R-CNN for the first time, the AP50 varied

between 21.7% and 27.7%. This difference in AP score is big and might be caused by diversity. To make sure it is not an accidentally created epistemic cause (meaning the images in the testset are completely different from the ones in the training set), the cross-validation was run two more times, each time with a different random mixture in each set. The three runs' outcomes differed a bit in the AP ranges but all of them showed similar gaps between the random sets as the first cross-validation.

To gain a clearer understanding of the unsatisfactory cross-validation results, specific experiments were conducted using YOLOv8n. The algorithm was trained exclusively on sunset or non-beach swimmer images and tested on daytime beach images to intentionally create an epistemic difference between the training and testing sets. The results were not satisfying. Figure 7.1 shows that the algorithm detected the swimmers in the back (but only with a low confidence score) while missing the swimmer in the front. While this happened in a few other images as well, the swimmer in the front was detected in most images so given a sequence it would be fine.

Figure 7.2 shows good detections of the swimmers on the right side of the image while no swimmers are detected on the left side. In the other images of this sequence, the detection results are similar to this image which is very dangerous (see Appendix A.2.1 for the full sequence). Overall, it can be stated that it is very important to have images of a similar condition at a similar place in the dataset to avoid FNs.



Figure 7.1: Dangerous missed detection



Figure 7.2: Swimmers on the left not detected

The only category with three different recordings at the same place during a similar light situation but on a different day is Sunset. Thus, it is possible to train the algorithm on a set of images that shows completely different people than the other sets. While two sets are almost in the exact same condition (Figure 7.3 and Figure 7.4), the third recording is very different (Figure 7.5).



Figure 7.3: Sunset recording 1



Figure 7.4: Sunset recoding 2



Figure 7.5: Sunset recording 3

When training YOLOv8n on the third set and testing on the first or second set, the algorithm is unable to detect the images correctly. When trained on the second set and tested on the thrid set, the algorithm is able to detect the swimmer in some images but mostly with a low confidence score (example Figure 7.6). However, by training on the first set and testing on the second set, the algorithm is able to detect most swimmers even though the confidence score is still very low (example Figure 7.7).



Figure 7.6: Correct detection in Sunset 3



Figure 7.7: All correct in Sunset 2

Going from the result of the cross-validation it is assumed that the epistemic difference between the images of the dataset is very big. Consequently, the dataset will be split similarly to [Varga et al., 2022] randomizing the frames while assuring that every situation is present in the training set, the validation set and the test set. This means that most situations in the test set will be in a similar way found in the training set as well (example Figure 7.8). It is clear that this means that the final model might not behave the same way on completely different images.

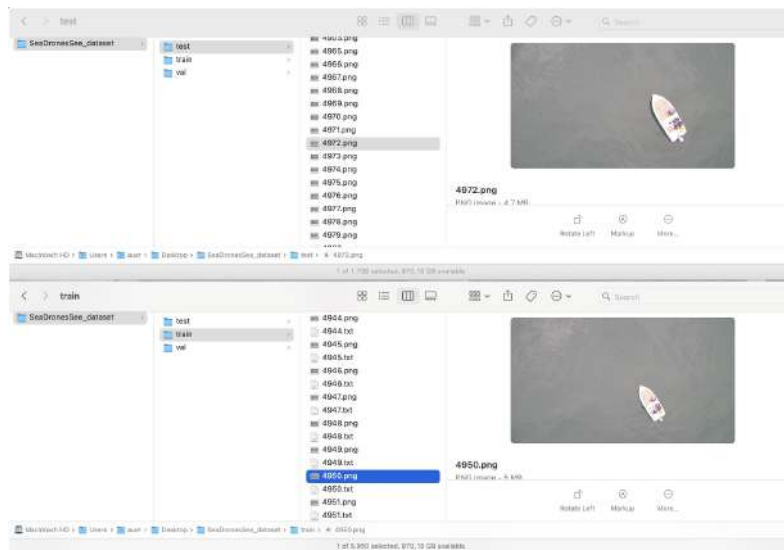


Figure 7.8: Example Dataset Split SeaDronesSee [Varga et al., 2022]

To check the final model's behaviour on completely different images, a partition of SeaDronesSee will be used. It needs to be acknowledged that this data will not represent the viewing angle of the ferry. However, testing on these images will help understand how the model behaves on any random image of a swimmer.

7.1.2 Dataset split

As explained in Section 2.2 the dataset is split into training, validation and test set. To determine how big the test set needs to be and based on this how big the training and validation set can be, some assumptions about the ferry are taken:

- The minimum distance (MD) between the ferry and the swimmer when detecting the swimmer is 70m.
- The ferry is travelling at its average speed (v) of 6knots (3 m/s) when the swimmer is detected in front of the ferry.

- The ferry travels 10m while making a decision (D) due to the computational power on board.
- The ferry takes one image per second (I_pS).
- The ferry needs 20m for an emergency stop (ES).
- The minimum distance (MS) between the swimmer and the ferry needs to be 10m at any given time.

Going from the taken assumptions the minimum amount of images available to detect a swimmer in front of the ferry while travelling is:

$$\begin{aligned}
& \frac{MD - D - ES - MS}{v} \times I_pS \\
&= \frac{70m - 10m - 20m - 10m}{3m/s} \times 1image/s \\
&= 10s \times 1image/s \\
&= 10images
\end{aligned} \tag{7.1}$$

The model should have no FNs close (70m distance) to the camera and less than one FP close (70m distance) to the camera per season. The route the ferry takes is a touristic route. Consequently, it only operated during summer. It operates up to 16 hours a day but only during daylight. On average it is expected, that the ferry to run 1500 hours per season. Since the ferry can record one image per second, the total amount of images per season is:

$$1,500 \frac{hours}{season} \times 3,600 \frac{seconds}{hour} \times 1 \frac{image}{second} = 5,400,000 \frac{images}{season} \tag{7.2}$$

For this thesis, it is not possible to test the algorithm on 5.4 million images. However, it is assumed that the probability of having a FN or FP will stay the same throughout the season.

As the final decision, if there is a swimmer in front of the ferry or not, is based on a sequence of 10 images (result (7.1)), the probability to have a FN or FP in a sequence will be calculated and scaled up to how many a FNs or FPs per season can be expected.

To guarantee good results in a sequence of 10 images, the test set consists of 28 sequences of 10 images. 19 sequences with situations from the new dataset and 9 sequences with situations from the Swimmer Dataset. The test set includes images with only one swimmer, two swimmers and a few with more than two swimmers. It has sequences taken during daylight as well as during sunset and features images with occluded swimmers as well as swimmers without a visible head.

Due to the input format of YOLOv8, the split between training and validation set had to be made by hand. It was attempted to randomize the split as much as possible, randomly picking 800 images for the validation set while making sure that all important situations are captured in images in the validation set as well as in the training set. The training set for YOLO consists of 2780 images.

For the training of Faster R-CNN the remaining data was split randomly by the algorithm. To obtain a similar training/validation set size, the data was divided into 77% training data (2862 images) and 23% validation data (718 images).

7.1.3 Data augmentation

When working with small datasets it is important to use data augmentation to improve the detection results. As described in Section 4.2.1 several types of data augmentation are used. The data is augmented by using the transforms function from PyTorch.

7.2 Faster R-CNN

Before training Faster R-CNN with the combined dataset, it was trained with the data from [Varga et al., 2022]. They offered their code as well as the exact split of the dataset and their results online. Based on that Faster R-CNN was trained and obtained an AP50 of 54.8% for the validation set which verifies the results given in Table 3.1 of 54.7%.

Proceeding from these good results, the same method was used as [Varga et al., 2022] to train Faster R-CNN with the combined dataset. As mentioned in Section 7.1.2 the data was split randomly between training and validation set using 77% of the data for the training set. During the training of Faster R-CNN, it achieved validation results of 26.1 for AP50 for the validation set. On average it took the model 21.5ms to process each test set image and make a prediction on a GPU 1070.

7.2.1 Optimizing Faster R-CNN

To optimize the detector's performance [Optuna](#) was used. Optuna tried the optimizer Adaptive Moment Estimation (Adam), Root Mean Squared Propagation (RMSprop) and Stochastic Gradient Descent (SGD) with a learning rate between 10^{-5} and 0.1. After several trials, Optuna was able to improve the performance of Faster R-CNN to 31.4 AP50 on the validation set using SGD and a learning rate of 0.002748.

As the cross-validation in Section 7.1.1 showed, the random subset used to validate the model can heavily influence the performance of the model. Using a different training validation set combination, the model improved its AP50 to 39.7 for the validation set with a P of 0.71 and an R of 0.95.

7.2.2 Performance on test set

To get a good ratio between FNs and FPs on the test set, Faster R-CNN was evaluated using several different Non-Maximum Suppression (NMS) thresholds. The NMS threshold is a parameter used to filter

out redundant bounding box predictions. It determines the minimum confidence score required for a bounding box to be considered during the suppression process.

To visualize better the confidence of the model for each bounding box, bounding boxes with a confidence score above 0.9 are marked in green, and bounding boxes with a confidence score below 0.9 are marked in yellow. As the figures in this chapter will show, most times Faster R-CNN detects swimmers with a confidence score above 0.9.

With a threshold of 0.01, the model had trouble with detecting people in a group during sunset. Most sunset images had a FN. As Figure 7.9 shows, both FNs are in a group with other swimmers. Since both times the swimmers around that person are detected with a high confidence score, the ferry would stop anyway.

On the other hand, there are 23 images with a FN that would be dangerous when not detected. For example in Figure 7.10 the ferry needs to stop immediately because there is a swimmer close by. Exactly this swimmer is not detected which is very dangerous.

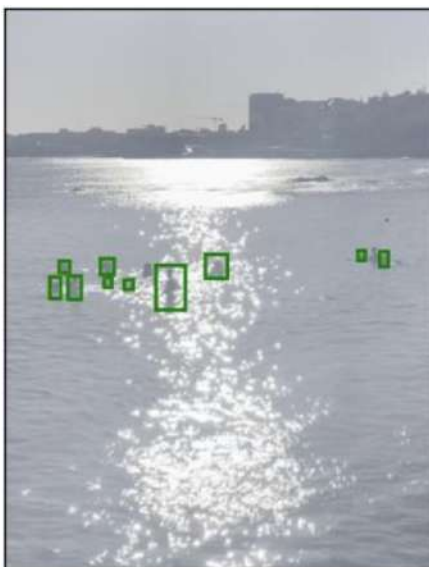


Figure 7.9: Example FNs during sunset

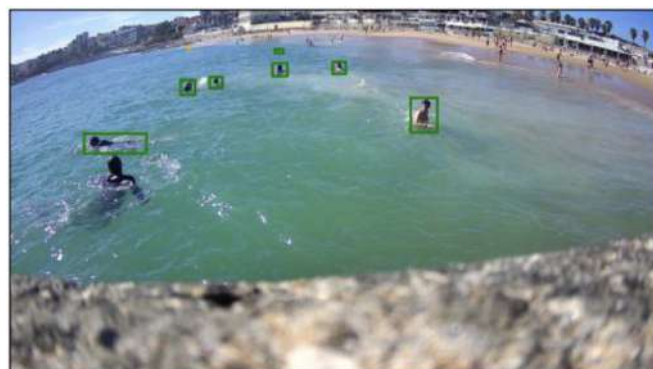


Figure 7.10: Example dangerous FN

If the goal was to have no FPs, this would have been a very good threshold considering there were almost no images with FPs. However, in this case, it is most important to detect every swimmer so as to have no FNs. To reduce the amount of FNs the NMS threshold was increased to 0.05.

With the new NMS threshold the amount of FNs was reduced. There are still several FNs in the group images like in Figure 7.9 but only four FNs that would have been very dangerous because the swimmer was alone on their part of the image. The swimmer on the right side of Figure 7.11 was not detected in 3 images in a row, while the swimmer on the left side (in the sunlight) of Figure 7.12 was detected in all the other images.



Figure 7.11: Example FN far from camera



Figure 7.12: Only dangerous FN close to camera

To check if it is possible to have no critical FNs the NMS threshold was increased to 0.1, 0.15 and finally 0.2. With a higher NMS threshold, it is possible to reduce the amount of FNs in the test set. However, the model never detected the three critical swimmers. At a threshold of 0.2 the amount of FPs increased a lot. Thus, it will not be helpful for the overall performance of the model to increase the threshold further.

After a close comparison of the results using a threshold of 0.1 and a threshold of 0.15, 0.15 is the better threshold: The test set had 4 FNs less than 0.1 and 2 FPs less. Anyhow, even with a threshold of 0.15, there are still the same FNs as presented in Figure 7.11 and Figure 7.12.

Nevertheless, as the goal is to detect a swimmer correctly in a sequence of images, this will be possible for all swimmers in the image sequences besides the FN in Figure 7.11. In this specific case, the third swimmer enters the scene shortly before the sequence in the test set ends. As the swimmer is not detected immediately but a few images after entering the scene, it can be expected that the algorithm would detect the swimmer correctly in further images as well.

Overall, there are 14 sequences of 10 consecutive images without any FN or FP.

7.3 YOLO

Ultralytics offers different YOLOv8 models. Since for the detection of swimmers in water both speed and correct results are important, the fastest YOLOv8n, as well as the most precise YOLOv8x, have been trained with the training and validation data presented in Section 7.1.2. Then both models made predictions on the test set which were manually checked and compared to decide which YOLOv8 model to improve further.

7.3.1 YOLOv8n

First, YOLOv8n was trained. During validation, the model achieved an AP50 of 34.8%, a P of 0.86, and an R of 0.66 with the validation set. It took the model 10.8ms to process each test set image and make a prediction. When manually checking the detection results on the test set, 21 images had an obvious FNs or FPs. Most incorrect detections showed one person detected as two individual swimmers (example Figure 7.13).

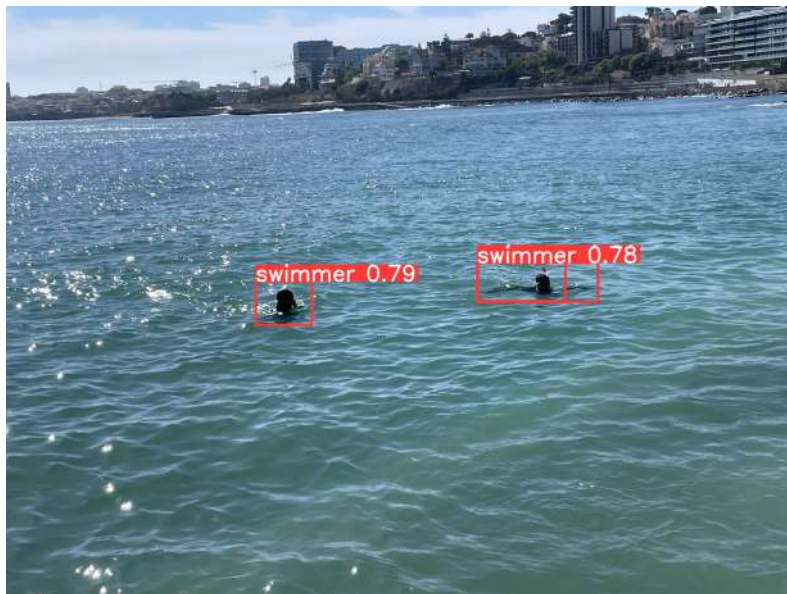


Figure 7.13: One person as two

YOLOv8n had only two FPs that are actually water. Figure 7.14 and Figure 7.15 show that both times the water is very crowded which does not correspond with the expected situation that the ferry will encounter a maximum of 2 people at the same time.

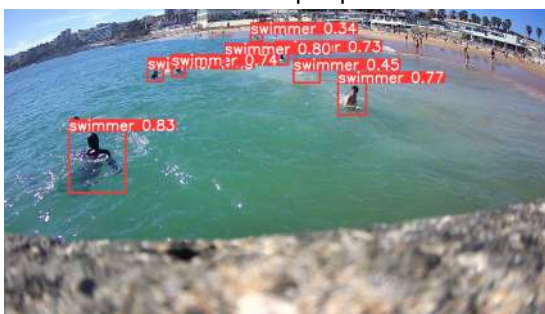


Figure 7.14: One false positive



Figure 7.15: Two false positives

In the predictions, there are eighth images with a FN. It happened three times in two consecutive images that the same person was not detected. To make sure that there are no FNs, it is important to compare several images of a sequence. This also includes comparing the position of each detected

swimmer, not only the amount of swimmers in the image.

For example Figure 7.16 shows three detections as the image beforehand in this sequence (see Appendix A.2.2 for the full sequence). However, one swimmer is detected twice while the other one is not detected. Based on the former images in the sequence, the algorithm can still assume correctly where the third swimmer is as he was detected correctly at the beginning of that sequence (while he is not visible for the camera later in the sequence).

Figure 7.17 shows that only one of the hands is detected as a human. This is already good as the human is detected. If there was only this swimmer, it would be an issue: the swimmer is estimated to be further away than it actually is. But in this situation, the ferry would stop for the swimmer closer to the camera anyways.



Figure 7.16: One false positive



Figure 7.17: Only one foot detected

Overall, there are 17 sequences of 10 consecutive images without any FN or FP.

7.3.2 YOLOv8x

According to Ultralytics, YOLOv8x is the most precise detector version of YOLOv8. During validation, the model achieved an AP50 of 40.4, a P of 0.84, and an R of 0.72 with the validation set. The model took 18ms to process each test set image and make a prediction. When manually checking the detection results on the test set, 17 images had obvious FNs or FPs. Half of the detections were FNs, and the other half FPs.

All of the images with a FP show the same person twice (example in Figure 7.13). Only in one image, the beach sand is detected as a swimmer. Since the ferry will know if there is water or land, FPs on land can be ignored. Detecting the same person as two people who are at the same spot, will not interfere with the ferry's actions towards that FP. Based on these two assumptions, YOLOv8x does not have any FPs.

There are nine images with a FN. Three times the FN occurs in two consecutive images. In Figure 7.18 and Figure 7.19 the person on the left is not detected while in Figure 7.19 and Figure 7.20 the

right person is not detected. This is a dangerous situation because the algorithm could assume that the swimmer moved under the surface from the right position to the left position. However, all three swimmers are detected in the rest of the sequence (given in Appendix A.2.3) thus it will be possible for the ferry to detect the three swimmers correctly.



Figure 7.18: Left person FN



Figure 7.19: Two FNs



Figure 7.20: Right person FN

Overall, there are 21 sequences of 10 consecutive images without any FN or FP.

7.3.3 Comparison YOLOv8n and YOLOv8x

Section 7.3.1 and Section 7.3.2 have shown that both YOLO algorithms work well. As expected YOLOv8x needs longer to predict on an image while being more precise.

Out of all test images, there are 10 images where both YOLO algorithms detected a FN or FP. On half of these images the prediction of both YOLOs was exactly the same. On the other five, they have missed a different swimmer or one had a FP that the other one did not.

All the FNs are in the same sequences for both YOLO algorithms. This shows that they share common difficulties. All FNs are in images with more than two swimmers so overall, both algorithms detect well swimmers in the range of one to two.

Based on the slightly better detection results of YOLOv8x and the fact that the ferry takes a second to process each image, it does not make sense to use YOLOv8n which is only 8ms faster in predicting.

7.3.4 Improving performance on test set

To improve the output of YOLOv8x, YOLOv8x was reevaluated using several different NMS thresholds as well as changing the confidence score. The initial NMS threshold used by Ultralytics to predict images is 0.7. Decreasing this threshold improved detection results as the amount of FNs decreases. Once the NMS threshold is below 0.25, there is a big increment in the amount of FPs. Most of them are two bounding boxes marking the same swimmer.

Another variable that influences if a bounding box becomes are prediction or not, is the confidence score. It describes how sure the model is that there is an object inside the bounding box. The initial value set by Ultralytics is 0.25. By reducing this value to 0.01, it was possible to see which is the lowest confidence score corresponding to a TP. In the case of the test set, the lowest confidence score for a TP

was 0.08 shown in Figure 7.21. However, allowing bounding boxes with this low of a confidence score to be counted as a detection also creates lots of FPs. Figure 7.22 shows an example of an image with several FPs.



Figure 7.21: Lowest confidence score for a TP



Figure 7.22: Several FPs with a low confidence score

Both Figure 7.21 and Figure 7.22 are from the same scene. As in most images (example Figure 7.22) the confidence score on TPs is a lot higher than 0.08. At the same time, many FPs have a confidence score below 0.2 thus it is a good compromise to set the confidence score to 0.2.

To improve the model's performance further [Lygouras et al., 2019] suggested to retrain the model with all images that had a FN or FP from the validation set. Accordingly, YOLOv8x was retrained with these images from the validation set. However, this did not improve the models' performance on the test set but increased the amount of FPs in the test set. Thus, the best model obtained before retraining with the validation set will be compared to Faster R-CNN.

7.4 Comparison Faster R-CNN and YOLOv8

As the cross-validation in Section 7.1.1 has shown that the used samples for training, validation and test set, heavily influence the model's performance, it is important to train YOLOv8 and Faster R-CNN with exactly the same data.

Faster R-CNN was retrained using the same dataset split as for YOLOv8 to ensure this is the case. With the same dataset split as YOLOv8 the AP50 increased a lot to 59.5%, P and R increased as well to 0.76 and 0.99. The overall performance of the model on the test set did not change significantly from the results presented in Section 7.2.

	Faster R-CNN	YOLOv8x
Detection speed	21.5ms	18ms
Validation set		
AP50	59.5%	40.4%
Test set		
AP50	46.4%	51.2%
Windows without mistakes	14	21
SeaDronesSee		
AP50	42.7 %	48.3%
Windows without mistakes	4 (+5)	4 (+2)

Table 7.2: Comparison Performance Faster RCNN YOLO

Table 7.2 shows a simple comparison of the performance of the two models. Since the epistemic difference in the combined dataset is so big (see Section 7.1.1), the test set featured images that are similar to the images used to train both models. To see how Faster R-CNN and YOLOv8x behave on completely new data, both models were also tested on 160 images from the SeaDronesSee dataset.

Table 7.2 gives the result of their performance on the test set, as well as on these 160 images from SeaDronesSee. The row FNs gives the number of images that have a FN while FPs gives the number of images that have a FP. If an image has both a FN and a FP, it will be counted twice. In the test set, each image sequence had 10 images. Thus, it is simple to determine how many sequences are without a mistake.

The images used from SeaDronesSee are not that clearly distributed. Thus, each different situation is counted as a correct sequence if the full sequence or at least 10 consecutive images from it are without a detection mistake. Faster R-CNN had an additional five times ten consecutive images that were without a mistake while YOLOv8x had additional two times ten consecutive images that were without a mistake.

7.4.1 Comparison on the test set

As given in Table 7.2 there are more FNs in the results from Faster R-CNN than in the results from YOLOv8x. Some of their mistakes are in the same sequence, some are in different sequences. Overall, YOLOv8x made more mistakes with swimmers close to the camera while Faster R-CNN made more mistakes with swimmers far from the camera. With swimmers at a medium distance, they had similar problems.

Table 7.3 gives examples of these FNs. Each image in this table with a FN represents a sequence with several FNs. As shown, the FNs at a medium distance are the same for both detectors. However, this is only the case in exactly this image. In the rest of the sequence it varies a bit in which images the models miss a swimmer.

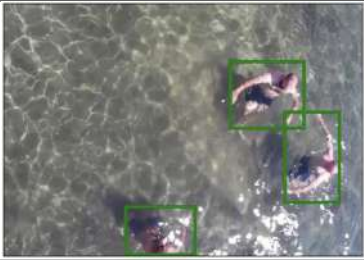





	Faster R-CNN	YOLOv8x
close to the camera		
at medium distances		
far away		

Table 7.3: Comparison Results on Test Set

Both detectors had FPs. However, if they appear in a group (example Figure 7.14) or if a person is detected twice (example Figure 7.13), they can be ignored since these FPs will not influence the detection results. Based on this it was stated in Section 7.3.2 that YOLOv8x does not have any FPs.

As shown in Table 7.2 most FPs from Faster R-CNN fall into this category too. The remaining six FPs will not influence the final detection results of each sequence as explained in Section 7.2. Thus, FPs do have a significant influence of the detection performance on either of the two models.

7.4.2 Comparison on images from SeaDronesSee

To better understand the detector's performance on completely new data, both models were tested on 160 images from SeaDronesSee. As given in Table 3.3 most images from the SeaDronesSee dataset had only very small swimmers. The 160 images chosen for this test only showed swimmers easily detectable to a human and without several boats in the image. Additionally, it is important to emphasize that these images mostly do not represent the viewing angle of the ferry.

Table 7.2 shows that Faster R-CNN had more FNs than YOLOv8x. At the same time Faster R-CNN had more sequences without mistakes. This means that in the remaining sequences mistakes from Faster R-CNN might affect the detector's performance more than in the results from YOLOv8x.

In contrast to the behaviour of the two models on the test set, here there was not a clear separation between the detection results. The only situation without any mistakes is this single swimmer shown in Figure 7.23.



Figure 7.23: Full sequence detected correctly

In all the other situations both models make mistakes. There is one scene where the UAV is first further away from the swimmer and then comes closer to him. In the far-away scene (Figure 7.24), Faster R-CNN has a FN in three images with two of them being consecutive images. Taking into account that the swimmer is detected in most of the images in the sequence, the model will still be able to detect the swimmer correctly.

Once the camera is closer to the swimmer (Figure 7.25), YOLOv8x has two images (with 12 correct images between them) with a FN. Since the gap between these two FNs from YOLOv8x is so big, it won't change the detection results in the sequence.



Figure 7.24: FN Faster R-CNN



Figure 7.25: FN YOLOv8x

In the remaining two situations, the mistakes are similar, even though Faster R-CNN has some more than YOLOv8x. Figure 7.26 shows a group scene with several swimmers. While these people float through the image, both models make mistakes with up to three FNs in the same image (image sequence from Faster R-CNN given in Appendix A.2.4). Nevertheless, it is a group scene thus, the ferry would stop for the remaining detected swimmers anyway.

The scene given in Figure 7.27 has dangerous results for both detectors. In this sequence of ten images, Faster R-CNN has six images where the left swimmer is not detected while YOLOv8x has seven images with the left swimmer not being detected. Since the boat is also visible on the right side

of the image, the ferry would most likely move to the left side to avoid the boat and the swimmer.



Figure 7.26: Scene with group of swimmers



Figure 7.27: Scene with two swimmers and a boat

Both algorithms had several images with a FP. As shown in Figure 7.27, there are scenes with a boat in the image. As expected in Section 5.4 both algorithms did detect the boat in the images from time to time. Independently if the boat is detected or not, it will not be counted as wrong as detecting boats is not part of this thesis.

Most of the FPs detected by Faster R-CNN are in group scenes. There is a sequence of six images with a FP at almost the same spot in every image. This sequence is given in Appendix A.2.4. This FP might be detected as a swimmer in the sequence. However, since there is an actual swimmer almost next to this FP, the ferry would choose to sail only a slightly larger distance to the actual swimmer, it would not drive differently than without the FP.

YOLOv8x never has more than 2 FPs in a sequence of 10 consecutive images. Thus, none of them will be detected as a swimmer when considering the full sequence.

7.5 Probability FN and FP per season

Based on the results achieved with Faster R-CNN and YOLOv8x, the probability of having FNs and FPs per season was calculated. As given in (7.2) there will be *5.4million* images per season thus, half a million windows of 10 images. Within these windows, the goal is to have a maximum of 1 FP and zero FNs. Consequently the probability of having a FP_w in the test set should be below 1.852×10^{-6} . While the probability of having a FN_w should be as close to 0 as possible.

To calculate the probability of a FN_w in the test set, first, all TPs and FNs per swimmer (bounding box) are counted. Coming from the values per swimmer, the TPs and FNs were determined as explained in Section 2.3.

Based on (7.1) a swimmer counts as detected if it is detected in several frames in a window of 10 images. To determine the probability of a swimmer being detected correctly, the true positive rate TPR or Recall (R) is calculated. The probability of a swimmer not being detected is called false negative rate

FNR. It is given as:

$$TPR = Prob(detection | Swimmer) = Recall = \frac{TP}{TP + FN} \quad (7.3)$$

$$FNR = Prob(-detection | Swimmer) = 1 - TPR \quad (7.4)$$

The probability for each window being a TP_w is calculated with:

- n , the number of detections in a time window
- N , the minimum of detection in a time window to count a swimmer as detected correctly
- W the number of frames per window (which is 10)

Based on these parameters, the formulas to calculate the TP_w and the FN_w are:

$$Prob(TP_w) = Prob(n \geq N | Swimmer) = \sum_{i=N}^W FNR^{W-i} \times avgTPR^i \times \frac{W!}{(W-i)! \times i!} \quad (7.5)$$

$$Prob(FN_w) = Prob(n < N | Swimmer) = \sum_{i=0}^n FNR^{W-i} \times avgTPR^i \times \frac{W!}{(W-i)! \times i!} \quad (7.6)$$

The probability of each window being a TP_w for Faster R-CNN and YOLOv8x given a certain N is represented in Table 7.4 belonging to the detection results of the test set and in Table 7.5 appertaining to the results of the unknown examples from SeaDronesSee.

To calculate the probability of having a FP in a window, theoretically, the amount of True Negatives (TNs) would be used. Since swimmers are not always the same size, the surrounding water cannot be divided into a certain amount of empty boxes to determine the TNs per window.

The ferry does not travel in a water area where humans normally swim. Thus, the ferry should normally not encounter a human. Referring to this assumption it was estimated that there would be a maximum of one swimmer per season found in the water. This implies, that there would never be more than one swimmer in the frame. Thus, FPPI (2.4) will be used to determine the probability of having a FP in the image, which is called false positive rate FPR .

Taking into account that all FP plus all TN must cover all GT negatives, the probability of having a TN is $1 - FPR$. It is called true negative rate TNR . The probability of a window having a FP_w is calculated similarly to the probability of a window being a TP_w :

$$Prob(FP_w) = Prob(n \geq N | noSwimmer) = \sum_{i=n}^W FPR^{W-i} \times TNR^i \times \frac{W!}{(W-i)! \times i!} \quad (7.7)$$

$$Prob(TN_w) = Prob(n < N | noSwimmer) = \sum_{i=0}^n FPR^{W-i} \times TNR^i \times \frac{W!}{(W-i)! \times i!} \quad (7.8)$$

The probability of each window having a FP_w for Faster R-CNN and YOLOv8x given a certain N is represented in Table 7.4 based on the detection results of the test set and in Table 7.5 appertaining to the results of the unknown examples from SeaDronesSee.

N	Faster R-CNN			YOLOv8x		
	$Prob(TP_w)$	$Prob(FN_w)$	$Prob(FP_w)$	$Prob(TP_w)$	$Prob(FN_w)$	$Prob(FP_w)$
1	1	$1.8 * 10^{-17}$	0.986	1	$1.2 * 10^{-14}$	0.505
2	1	$8.3 * 10^{-15}$	0.910	1	$2.8 * 10^{-12}$	0.144
3	1	$1.7 * 10^{-12}$	0.731	1	$3.0 * 10^{-10}$	0.026
4	1	$2.2 * 10^{-10}$	0.477	0.999	$1.9 * 10^{-8}$	0.003
5	0.999	$1.8 * 10^{-8}$	0.241	0.999	$8.1 * 10^{-7}$	$2.7 * 10^{-4}$
6	0.999	$9.8 * 10^{-7}$	0.091	0.999	$2.3 * 10^{-5}$	$1.6 * 10^{-5}$
7	0.999	$3.8 * 10^{-5}$	0.025	0.999	$4.6 * 10^{-4}$	$6.6 * 10^{-7}$
8	0.999	0.001	0.004	0.994	0.006	$1.8 * 10^{-8}$
9	0.982	0.018	$4.9 * 10^{-4}$	0.941	0.059	$2.8 * 10^{-10}$
10	0.807	0.193	$2.5 * 10^{-5}$	0.66	0.339	$2.1 * 10^{-12}$

Table 7.4: Probability for a window detection results - test set

N	Faster R-CNN			YOLOv8x		
	$Prob(TP_w)$	$Prob(FN_w)$	$Prob(FP_w)$	$Prob(TP_w)$	$Prob(FN_w)$	$Prob(FP_w)$
1	1	$1.5 * 10^{-10}$	0.772	1	$7.0 * 10^{-11}$	0.509
2	0.999	$1.3 * 10^{-8}$	0.409	0.999	$6.6 * 10^{-9}$	0.147
3	0.999	$5.2 * 10^{-7}$	0.148	0.999	$2.8 * 10^{-7}$	0.027
4	0.999	$1.2 * 10^{-5}$	0.038	0.999	$7.2 * 10^{-6}$	0.003
5	0.999	$1.9 * 10^{-4}$	0.007	0.999	$1.2 * 10^{-4}$	$2.9 * 10^{-4}$
6	0.998	0.002	$8.6 * 10^{-4}$	0.999	0.001	$1.7 * 10^{-5}$
7	0.985	0.015	$7.6 * 10^{-5}$	0.989	0.011	$7.2 * 10^{-7}$
8	0.922	0.078	$4.4 * 10^{-6}$	0.936	0.064	$1.9 * 10^{-8}$
9	0.720	0.280	$1.5 * 10^{-7}$	0.750	0.250	$3.2 * 10^{-10}$
10	0.333	0.667	$2.4 * 10^{-9}$	0.363	0.637	$2.4 * 10^{-12}$

Table 7.5: Probability for a window detection results - SeaDronesSee examples

YOLOv8x has better probabilities for not having a FP in both tables while the probability of having a FN is a little higher in the test set than for Faster R-CNN. This is caused by 80 FN very far away from the boat which were not counted in the comparison in Section 7.4 as they do not influence the behaviour of the ferry considering that the swimmer close to the camera is detected correctly (see full sequence in Appendix A.2.3).

As mentioned earlier, the goal is for the ferry to at most have one FP per year. Therefore the value for $Prob(FP_w)$ must be below $1.85 * 10^{-6}$. Choosing an N equal to or higher than seven will guarantee that. To get as little FNs as possible as well, the best option is to choose $N = 7$. Based on the values from Tables 7.4 and 7.5 and the assumption for the ferry to encounter a maximum of one swimmer per year SpY , the number of years until a swimmer will be missed can be calculated:

$$\frac{1SpY}{Prob(FN_w) \text{ for } N = 7} = 2174 \text{ years when knowing similar images} \quad (7.9)$$

$$= 91 \text{ years when not knowing similar images} \quad (7.10)$$

Referring to (7.9) it will take 2174 years until a person is at risk if there are similar images in the training set. On the other hand, according to (7.10) it only takes 91 years until a person may be harmed by the ferry if there are no similar images available in the dataset. Based on these two numbers, having similar photos to what the ferry is seeing in the dataset will be very important.

7.6 Implementation model

The comparison of the detection results of Faster R-CNN and YOLOv8 in Section 7.4 has shown that YOLOv8x has less FNs than Faster R-CNN plus less or none FPs. With the faster detection speed, it is at clear advantage to produce good detection results for swimmers faster.

By combining the initial problem presented in Section 1.2 with the probabilities for FNs and FPs calculated in Section 7.5, it is highly probable that when using the model developed with YOLOv8x, the ferry will avoid hitting anyone in the water under the specified assumptions. Additionally, the ferry is expected to make stops no more than once per season. These results suggest that the YOLOv8x model provides a reliable and safe solution for detecting swimmers in front of the ferry, mitigating potential accidents while minimizing unnecessary interruptions during its operations.

8

Conclusion

Contents

8.1 Outcome	68
8.2 Future work	69

To finalize this thesis, the outcome will be summarized and proceeding from there some ideas for future improvement will be given.

8.1 Outcome

The aim of this master thesis was to create a model that automatically detects swimmers in water. As presented in Chapter 3 there are only a few existing datasets that feature swimmers. Since most of these datasets are based on UAVs and do not represent the view of a ferry on the water, a new dataset was created.

As presented in Chapter 6 first the existing two datasets SeaDronesSee [Varga et al., 2022] and Swimmer dataset [Lygouras et al., 2019] were analyzed to see how important it is to cover different distances between the camera and the swimmer, different weather situations, and the variety of swimmers in the dataset. The results emphasized the need for additional images from the ferry's perspective, including coverage of swimmers occluded by waves and images captured during sunset/sunrise.

The newly created dataset addresses these requirements by including images taken during sunset on two different days. These images feature both groups of swimmers and individual swimmers positioned beside and in front of the reflected sun. Furthermore, the dataset includes various images featuring partially and fully occluded swimmers. Additionally, there has been an augmentation of swimming styles in the dataset, with images of freestyle swimming and diving captured of a commercial diver and an athlete swimmer. Lastly, a few images with the thermal camera presented in Section 6.1.2 were captured but as the camera uses IR to create the images, the visibility of the swimmer did not increase thus, these images were not included in the final dataset.

The cross-validation presented in Section 7.1.1 showed that there is a big epistemic difference in the dataset as it now features images of several different situations. Thus, the dataset was split similarly to [Varga et al., 2022] which decreases the originality of the test set but allows the use of all the situations in the new dataset to train the model.

To still get a good feeling of how Faster R-CNN and YOLOv8x perform on completely new data of swimmers, 160 images from SeaDronesSee were used to test the models' performance after being trained. A comparison of these detection results to the test set, emphasized again how important it is to have similar images in the training set as the ferry will encounter while travelling as well as the influence of the camera angle on the swimmer. The swimmer from SeaDronesSee with a camera angle similar to the one of the ferry (and thus the training set) was always detected correctly (example in Figure 7.23).

Independently of detection results on single images, it is important to see how the models perform on a sequence of images as the ferry will not take its decisions based on a single image but always compare results of a sequence of images. Overall, detection results in image sequences were very

good with only one sequence containing a dangerous situation where a swimmer was rarely detected in the sequence.

As it was not possible to have a test set with as many images as the ferry will take during one season, the probability of Faster R-CNN and YOLOv8x having a FN_W or FP_W per season was determined in reference to the outcome of the test set. Using YOLOv8x and data similar to the training set, it is possible to have less than one FP_W per season while still guaranteeing that nobody will be harmed by the ferry over the course of 2000 years

With this outcome, we are able to fulfil the goal of this thesis to create a model that safely detects swimmers in open water and improves the operational safety of an autonomous ferry in the future.

8.2 Future work

As shown in Section 7.4 and Section 7.5, it is very important to have images similar to the test set in the training set. As it was not possible due to the limited time of this thesis to take images with swimmers directly from the ferry at Helsinki where it travels, it would be very important to create these images and finetune YOLOv8x with them to increase detection results in this specific travel area.

For the current probability calculation to classify a window as a swimmer, the position of the swimmer was not taken into account. Since the FPs in a sequence would often appear in random places, comparing the spot where a swimmer is detected in different images will decrease the probability of having a FP_{window} . [ByteTrack](#) could be used for to track the swimmer. This would enable us to decrease the probability of missing a swimmer in front of the ferry as well by needing fewer correct frames per window, as well as making sure, that FNs as presented in Section 7.3.1 Figure 7.15 will be less likely to happen.

Additionally, the company is planning to increase the fps rate of the final solution to have more images for the same distances. Thereby, increase the chances of detecting a swimmer. Together with implementing a safety margin that the ferry will already start slowing down at a certain amount of detections without fully stopping, the FN rate especially for completely unknown situations will drop even lower.

Bibliography

- [Bappy and Roy-Chowdhury, 2016] Bappy, J. H. and Roy-Chowdhury, A. K. (2016). Cnn based region proposals for efficient object detection. In *2016 IEEE International Conference on Image Processing (ICIP)*.
- [Benderius et al., 2021] Benderius, O., Berger, C., and Blanch, K. (2021). Are we ready for beyond-application high-volume data? the reeds robot perception benchmark dataset. *Computer Vision and Pattern Recognition*.
- [Berger, 2013] Berger, J. O. (2013). *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media.
- [Cafarelli et al., 2022] Cafarelli, D., Ciampi, L., Vadicamo, L., Gennaro, C., Berton, A., Paterni, M., Benvenuti, C., Passera, M., and Falchi, F. (2022). Mobdrone: A drone video dataset for man overboard rescue. In *Image Analysis and Processing – ICIAP 2022*.
- [EMSA, 2021] EMSA (2021). Annual overview of marine casualties and incidents. Technical report, European Maritime Safety Agency (EMSA).
- [Ge et al., 2021] Ge, Z., Liu, S., Wang, F., Li, Z., and Sun, J. (2021). Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.
- [Images, 2020] Images, G. (2020). istock. <https://www.istockphoto.com/pt/fotos/water-reflection-below-human-hand>. Last accessed 31 December 2022.
- [Inc, 2021] Inc, G. (2021). Gv-tdr2700 series. <https://www.geovision.com.tw/product/GV-TDR2700%20Series>. Last accessed 27 December 2022.
- [Jacquelin et al., 2022] Jacquelin, N., Vuillemot, R., and Duffner, S. (2022). Detecting swimmers in unconstrained videos with few training data. In *International Workshop on Machine Learning and Data Mining for Sports Analytics*.

- [Janai et al., 2020] Janai, J., Güney, F., Behl, A., and Geiger, A. (2020). Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*.
- [Jiao et al., 2019] Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., and Qu, R. (2019). A survey of deep learning-based object detection. *IEEE Access*.
- [Koech, 2020] Koech, K. E. (2020). Object detection metrics with worked example. *Towards Data Science*.
- [Kohavi et al., 1995] Kohavi, R. et al. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*.
- [Leira et al., 2015] Leira, F. S., Johansen, T. A., and Fossen, T. I. (2015). Automatic detection, classification and tracking of objects in the ocean surface from uavs using a thermal camera. In *2015 IEEE aerospace conference*.
- [Li, 2021] Li, W. (2021). Infrared image pedestrian detection via yolo-v3. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*.
- [Lin et al., 2017] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*.
- [LLC, 2021] LLC, T. F. (2021). Flir e4. <https://www.flir.eu/products/e4/>. Last accessed 27 December 2022.
- [Lygouras et al., 2019] Lygouras, E., Santavas, N., Taitzoglou, A., Tarchanidis, K., Mitropoulos, A., and Gasteratos, A. (2019). Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations. *Sensors*.
- [Minaee et al., 2022] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D. (2022). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Nguyen et al., 2017] Nguyen, D., Hong, H., Kim, K., and Park, K. (2017). Person recognition system based on a combination of body images from visible light and thermal cameras. *Sensors*.
- [opencv, 2022] opencv (2022). Cvat. <https://github.com/opencv/cvat>. Last accessed 31 October 2022.
- [Prasad et al., 2017] Prasad, D. K., Rajan, D., Rachmawati, L., Rajabally, E., and Quek, C. (2017). Video processing from electro-optical sensors for object detection and tracking in a maritime environment: A survey. *IEEE Transactions on Intelligent Transportation Systems*.

- [PyTorch, 2017] PyTorch (2017). fasterrcnn_resnet50_fpn. https://pytorch.org/vision/main/models/generated/torchvision.models.detection.fasterrcnn_resnet50_fpn.html#torchvision.models.detection.FasterRCNN_ResNet50_FPN_Weights. Last accessed 04 July 2023.
- [Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Redmon and Farhadi, 2018] Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.
- [Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Sandler et al., 2018] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Schubert et al., 2004] Schubert, C. M., Leap, N. J., Oxley, M. E., and Jr., K. W. B. (2004). Quantifying the correlation effects of fused classifiers. In *Signal Processing, Sensor Fusion, and Target Recognition XIII*.
- [Shin et al., 2020] Shin, H.-C., Lee, K.-I., and Lee, C.-E. (2020). Data augmentation method of object detection for deep learning in maritime image. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*.
- [Terven and Cordova-Esparza, 2023] Terven, J. and Cordova-Esparza, D. (2023). A comprehensive review of yolo: From yolov1 and beyond. *arXiv preprint arXiv:2304.00501*.
- [Todabasi, 2022] Todabasi, A. (2022). shiny sparkles sun reflection ocean. <https://depositphotos.com/339356374/stock-video-shiny-sparkles-sun-reflection-ocean.html>. Last accessed 31 December 2022.
- [Trick and Rothkopf, 2022] Trick, S. and Rothkopf, C. (2022). Bayesian classifier fusion with an explicit model of correlation. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*.
- [Ultralytics, 2023] Ultralytics (2023). Ultralytics yolov8. <https://github.com/ultralytics/ultralytics>. Last accessed 04 July 2023.

- [Varga et al., 2022] Varga, L. A., Kiefer, B., Messmer, M., and Zell, A. (2022). Seadronessee: A maritime benchmark for detecting humans in open water. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.
- [Wu et al., 2020] Wu, D., Lv, S., Jiang, M., and Song, H. (2020). Using channel pruning-based yolo v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Computers and Electronics in Agriculture*.
- [Xu and Goodacre, 2018] Xu, Y. and Goodacre, R. (2018). On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of analysis and testing*.
- [Yassine et al., 2020] Yassine, B., Larbi, G., and Hicham, L. (2020). Human detection in surveillance videos using mobilenet. In *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*.
- [Zhang et al., 2020] Zhang, S., Xie, Y., Wan, J., Xia, H., Li, S. Z., and Guo, G. (2020). Widerperson: A diverse dataset for dense pedestrian detection in the wild. *IEEE Transactions on Multimedia*.
- [Zhao et al., 2019] Zhao, Z.-Q., Zheng, P., Xu, S.-T., and Wu, X. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*.
- [Zhou et al., 2019] Zhou, X., Wang, D., and Krähenbühl, P. (2019). Objects as points. *arXiv preprint arXiv:1904.07850*.



Appendix A

A.1 Proofs Generalisation

The dataset for this thesis was created based on certain assumptions (Section 1.2). Based on the analysis of the two existing datasets in Chapter 7 some of the initial generalizations had to be proven to not lose important data. The experiments taken to validate the generalizations are provided in this appendix.

A.1.1 Floater vs Swimmer

In the SeaDronesSee dataset, the authors divide between swimmers and floaters (swimmers wearing a lifejacket). We will not have this distinction because it does not matter for our purpose. To ensure that grouping them together will not influence the detection rate of the model, I trained a FasterRCNN model once with only floaters, testing it on swimmers and once with only swimmers, testing it on floaters.

When training on floaters and testing on swimmers, all of the validation set images which only contained swimmers were detected correctly. In the test set some images contained false positives but all

of them were far away from the camera (example see Figure A.2) thus only little influence on the ship's behaviour. Figure A.1 shows that the model performed very well.

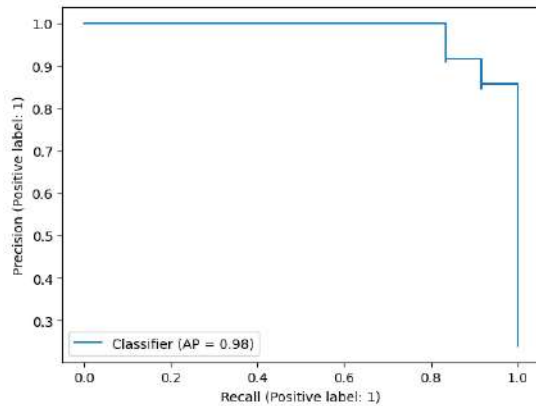


Figure A.1: Training floaters



Figure A.2: Example far away false positive

When training with swimmers and testing with floaters, the overall result was worse than in the opposite direction, see Figure A.3. However false positives were either reflections of people sitting on a boat (example Figure A.4) or humans that were classified as two swimmers while both body parts belonged to the same floater (similar to Figure 5.20). The only false negatives were images of floaters far away and seen from a drone.

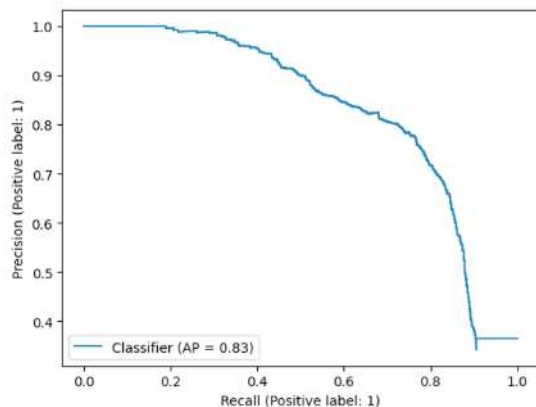


Figure A.3: Training swimmers



Figure A.4: Example far away false positive

Overall there is no big difference in the performance of the algorithm if the person is wearing a life jacket or not.

A.1.2 Object in the image

The testset of images with swimmers and objects featured only 78 images. However, none of these images had a false negative. In very few the swimmer was predicted in a wrong size which then included the object as well (example Figure A.5). More common was that the object was detected beside the swimmer without any influence on the detection of the swimmer. Especially when the detection is outside

of the water as in Figure A.6 it does not influence the behaviour of the ferry at all since it is already given that the ferry knows which part of its surrounding is water and which is shore.



Figure A.5: Swimmer and Object detected as one

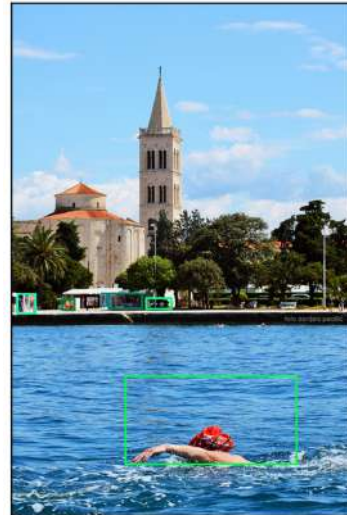


Figure A.6: Detections on shore

A.1.3 Animal in the image

When training the model on images with only swimmers and then testing on images with swimmers and animals (in total 258 test images), the results first showed lots of false positives. However out of 150 images with false positive detections, only six images actually showed false positives. In all the other 144 images the model detected an animal that was not marked in the ground truth since it is not a swimmer (example Figure A.7).

There were a few images where the model only detected the animal but not the swimmer (example Figure A.8). All false negatives were swimmers far away from the camera. Taking into account that the animal was detected, the ferry would already know that there is something in the water. Swimmers close to the camera were all detected. Therefore it is assumed that if the swimmer gets closer to the camera they would have been detected too despite the animal.



Figure A.7: Animal and swimmer detected



Figure A.8: Animal detected but not swimmer

A.2 Detection results in sequence

Since the final detection of a swimmer always depends on the results in an image sequence, this section will present the full image sequence of a detector.

A.2.1 Cross-Validation

The full sequence talked about in Section 7.3.1 is given here. As mentioned the model has difficulties to detect swimmers on the left side.



Figure A.9: Image 163953 071



Figure A.10: Image 163954 135



Figure A.11: Image 163955 227



Figure A.12: Image 163956 319



Figure A.13: Image 163957 412



Figure A.14: Image 163958 477



Figure A.15: Image 163959 574



Figure A.16: Image 164000 648



Figure A.17: Image 164001 742



Figure A.18: Image 164002 813

A.2.2 YOLOv8n

The full sequence talked about in Section 7.3.1 is given here. The images come exactly like presented here as a sequence in the Swimmer dataset. However, estimated by the naming of the images there are most likely some images missing in comparison to the original recording.



Figure A.19: Image 6482



Figure A.20: Image 6484



Figure A.21: Image 6489



Figure A.22: Image 6494



Figure A.23: Image 6495



Figure A.24: Image 6498

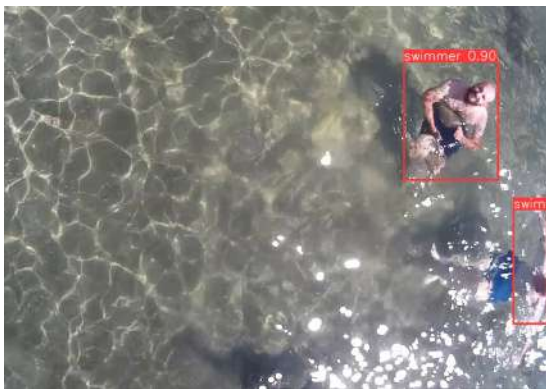


Figure A.25: Image 6505



Figure A.26: Image 6507



Figure A.27: Image 6509



Figure A.28: Image 6511

A.2.3 YOLOv8x

The full sequence talked about in Section 7.3.2 is given here. Since these images are taken with the RGB camera presented in Section 6.1.1, the image name refers to the time when the image was recorded.

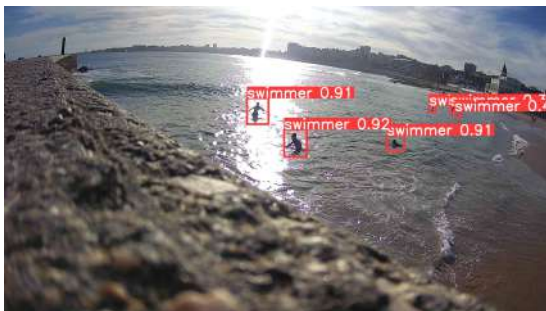


Figure A.29: Image 183539 481



Figure A.30: Image 183540 556



Figure A.31: Image 183539 481



Figure A.32: Image 183542 725



Figure A.33: Image 183543 828



Figure A.34: Image 183544 907



Figure A.35: Image 183545 983



Figure A.36: Image 183547 083



Figure A.37: Image 183548 171



Figure A.38: Image 183549 274

A.2.4 Faster R-CNN

As described in Section 7.4.2 there is a sequence with several FPs at the same spot. The image names are copied from the name given by [Varga et al., 2022].



Figure A.39: Image 2104



Figure A.40: Image 2105

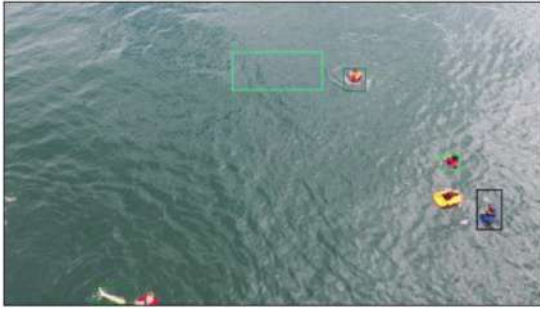


Figure A.41: Image 2106

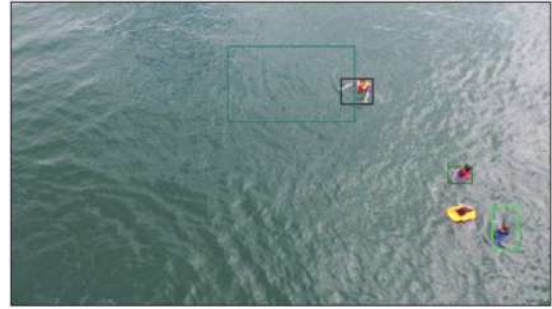


Figure A.42: Image 2107

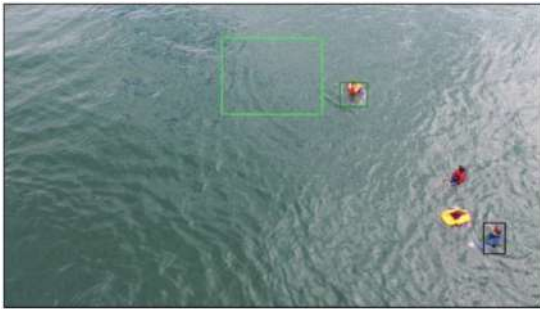


Figure A.43: Image 2108

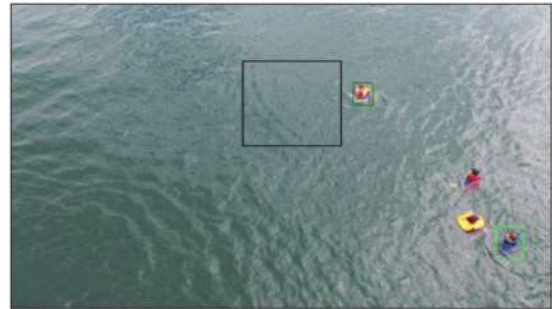


Figure A.44: Image 2109



Figure A.45: Image 2110

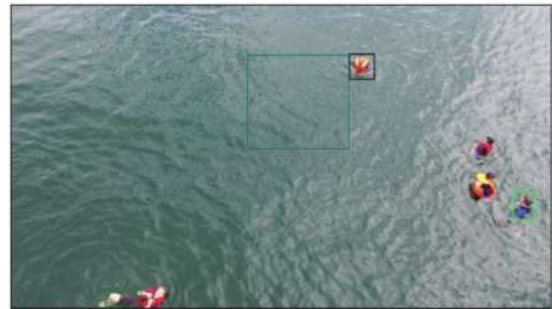


Figure A.46: Image 2112



Figure A.47: Image 2113



Figure A.48: Image 2114

