NHL STENDEN
computer vision & data science

Transfer Learning for Semantic Ship Defect Segmentation

NHL Stenden Lectoraat in Computer Vision & Data Science

Alexander de Haan, Ruben Campuzano Hernandez

Supervisors: Klaas Dijkstra, Henry Maathuis

Abstract—The inspection of ship-board structures by humans is a time-consuming and expensive activity, creating the need to search for better solutions. We have realized that as the implementation of segmentation on common problems becomes more realistic, there is an opportunity to apply this to the detection of ship defects. In this study, we investigated how transfer learning and data augmentation interact. work in the segmentation of defects. Semantic segmentation models, specifically U-Net, are trained using five data sets in total.The five datasets used were manually created by combining different percentages of a concrete crack dataset and 73 ship break and rust images. We also studied how the performance of these models changes when applying data augmentation (grayscaling, rotating, as well as random contrast).To evaluate each experiment, we used the IoU, the precision, the recall, and the F1-score. The results of the experiments indicate that without transfer learning, a precision above 50% can be achieved. Recall, however, would be too low for practical use. Using transfer learning, recall can be increased above 90% at the cost of precision. or degrade performance This research shows that a U-Net model with transfer learning can segment damage in ships. It would be of interest to see if few-shot learning can be used to improve the precision of segmentation.

Index Terms—[Transfer Learning, Data Augmentation, Semantic Segmentation, U-Net, Ship Defect]

1 Introduction

The United Nations Conference on Trade and Development (UNCTAD) reports that over 80% of the volume of international trade in goods is carried by sea, and that percentage is even higher for most developing countries [1]. Nowadays, companies use big containers on huge ships for transporting all kinds of products, but in the past, they used big wooden boats to move their goods. These methods have evolved with the new technologies, but that fact does not mean that the materials used on those huge ships are perfect. They also get old and suffer breaks because of the water conditions.

The maintenance of these ships is important for the functioning of the worldwide trading system. So all ships need to be checked to prevent major damage. Most of the time, all the maintenance is done manually, by having a specialist come to the ship to check where the damage was. With the development of Internet technology and the arrival of the era of big data, most companies are focusing their attention on new artificial intelligence to find damage instead of manual inspection.

Corrosion and rust are common damage in ships as a result of aging. To reduce the reparation costs, these companies need an efficient way to detect problems before they get worse. Being able to repair ships does not only reduce economic losses, it can also prevent catastrophic accidents and pollution of the marine environment.

For example, the bottom of the ship is usually protected by paint, so that part is theoretically insulated. However, if the paintwork gets scratched, it can result in two different metals being submerged in electrolyte and short-circuiting, so an electrical current flow will immediately begin to flow. This situation is common. Thus, to prevent any kind of catastrophic marine accident, it is of great importance to carry out damage detection.

The main goal of this work is to automate the detection of damage to the hull of ships. To achieve this, a segmentation algorithm, namely a U-Net network [2], will be trained on relevant annotated images, details of which can be found in section 3.1.

This research paper addresses the following research question: *Can Artificial Neural Network based image segmentation be implemented to detect damages on the hull of ships?*
This main research question can be decomposed into several questions:

- *Can a U-Net image segmentation network achieve good performance in segmenting damages on ships? Where good performance is based primarily on the precision and recall, and a score above 0.8 will be considered a good score.*
- *Can a similar, but not directly related, dataset be used to improve model performance?*
- *Can data augmentation help improve model performance?*

2 State of the art

There are various articles about how to detect damages on different surfaces using Image Segmentation with U-Net models.
In [3], a convolutional neural network is implemented to detect and recognise corrosion damage in ships. They propose an overlap-scanning sliding window algorithm. This is used with AlexNet. AlexNet[4] is a Deep Convolutional Neural Network designed to classify images in the ImageNet dataset.

The model helps to improve the detection of corrosion spots with different till sizes. They conclude that the method proposed was valid for most cases, but due to fewer images under the conditions of weak light intensity, weak blurriness, and shade, the AlexNet model's generalization ability was not high for such images.

More recently, in [5], a U-Net model was implemented to detect cracks in steel structures. To do this, the authors built an automated workflow to assess the fracture mechanism from crack images. From the procedures, it can be seen how they separate the breaks into three different groups due to their size. In the paper, it is also explained how U-Net works. They use an encoder based on the well-researched VGG-19 CNN [6] classifier and a decoder that mirrors the encoder.

The authors conclude that although there was a higher false-positive rate in the testing dataset, the study had good performance in identifying the pixel-level crack location considering

Alexander de Haan is a Computing Science student at the NHL Stenden University of Applied Sciences, E-mail: alexander.de.haan@student.nhlstenden.com.

Ruben Campuzano Hernandez is a Computing Science student at the NHL Stenden University of Applied Sciences, E-mail: ruben.campuzano.hernandez@student.nhlstenden.nl.

Klaas Dijkstra is an associate lector at the NHL Stenden Lectoraat in Computer Vision & Data Science, E-mail: maya.aghaei.gavari@nhlstenden.com.

Henry Maathuis is a researcher at the NHL Stenden Lectoraat in Computer Vision & Data Science, E-mail: henry.maathuis@nhlstenden.com

different kinds of images (different sources, different scales and resolution, and some variety in the type of break).

In [5], the researchers used 200 corrosion images and 500 crack images with their pixel-level annotations. The paper explains how the researchers implemented two options for the data treatment, squashing and cropping, to establish images to a constant size (224x224). The primary difference between these two methods was that with cropping, you increase the size of the dataset by more than 50 times the initial size. Each experiment is evaluated by the metrics explained in the paper (accuracy, precision, recall, and F1-Score).

In the experiments, they noticed that cropping images was better than the squashing method. That is because it has better results in the detection of corrosion. But they also discovered that the cropping method is not that good for break detection. So, to solve that problem, they propose a new method in which they randomly drop some cropped backgrounds, creating a new dataset with much relevant information. The Background Data Drop Rate (BDDR) is the percentage of useless cropped background images that are deleted. The paper concludes by suggesting that the best way to get better results is to increase the BDDR parameter.

In [7], researchers attempted to apply transfer learning to biomedical image segmentation. Transfer learning uses two datasets to train a model. One large dataset but not entirely representative of the target data, usually a dataset used in previous research. The second dataset is one that is more representative of the problem. Using this method, the hope is that researchers do not need to annotate a large, precisely representative dataset and instead only need to annotate a smaller dataset whilst being able to reuse datasets from previous research.

The usage of transfer learning in this research managed to reduce classification errors by up to 60%.

A downside of transfer learning is that the external dataset and the target data need a high degree of similarity. If the dataset is not sufficiently similar, it will result in a worse model rather than a better one.

The application of transfer learning in this paper might increase the accuracy of the model, especially considering the available dataset for this research is small.

3 Materials and Methods

This section elaborates on the datasets used throughout the paper and explains the model architecture used in this paper. Furthermore, this section explains the tiling used during preprocessing. Additionally it will explain the means by which the results are evaluated.

3.1 Dataset

3.1.1 Ship defect dataset

This dataset [8] contains breaks in a metal surface and hull rust images. Each image has a different size between 2359px x 1582px and 674px x 446px. The images have different resolutions, but all of them are RGB images. Every image is classified by the color of the image. They are divided into 6 folders (white and brown, brown and green, dark images, bright images), but for the training they are all mixed. The dataset is split into three sets, each holding 50 for training, 13 for validation, and 10 for testing. In the Appendix B.1 there are some examples of the images.

3.1.2 Concrete Breaks

This dataset contains concrete images [9] split into 44 for training, 15 for validation, and 6 for testing. The images have a size of 448px x 448px with a low resolution. There is a substantial difference between the types of damage from this dataset compared to the other one. But the main purpose of using this dataset is to see if the rust on a metal surface can turn into a break. Anyway, the size of these datasets is not big enough for training a good model. In the Appendix B.2 there are some examples of the images.

3.1.3 Mixed Dataset

In an attempt to improve the performance of the model, we combined the two existing datasets, varying proportions of each dataset, resulting in 5 different ones. The idea is to get a balanced base where the model can learn all the types of damage. The resulting dataset contains 80 training and 30 validation images. It has to be noted that each kind of image has a different type of annotation. The crack images have a much more precise annotation than the ship defect annotations. That is why the learning process for the model is not the best. You can see the proportions of each type of dataset in the Appendix B.3

These are some examples of images and their annotations:

(a) Concrete Break Image

(b) Ship Defects Image

Fig. 1: Images and Annotations from the datasets

3.2 Data Augmentations

Data augmentation (DA) [10] [11] refers to techniques to expand the quantity of data via adding slightly altered copies of the current data or creating new synthetic data from existing data. It acts as a regularizer and it usually helps to reduce over fitting. The purpose of the augmentations used is to make datasets more similar to one another in an attempt to improve model performance. Albumentations is the library utilized in the case presented [12]. It provides a single interface for working with various computer vision tasks like classification, semantic segmentation, instance segmentation, object identification, and posture estimation.

Furthermore, image thresholding was applied such that the intensities of the images were 0 for the background and 1 for the defect.

3.3 Model architecture

One of the most commonly used architectures in semantic segmentation is U-Net. The reason for using that model is that we need to convert feature maps into a vector but also reconstruct an image from this vector. This is a huge task because it is much tougher to convert a vector into an image than vice versa. The whole idea of U-Net is to solve this problem.

U-Net uses the same feature maps that are used for contraction to expand a vector to a segmented image. This preserves the structural integrity of the image, which reduces distortion.

The used baseline architecture of U-Net has 64 start filters, 5 depth layers, and a latent vector of size 1024. During the experimentation section, these parameters can be modified. This architecture can be divided into 2 parts:

Encoder It consists of repeated applications of 3x3 convolution layers. Each convolution is followed by a ReLU nonlinearity and batch normalization. The model uses a 2x2 max pooling operation with a kernel size of 3 to reduce the spatial dimensions.

Decoder This part consists of up sampling the feature maps followed by a 2x2 transpose convolution. The model also has a 3x3 convolutional layer followed by a ReLU. Finally, a 1x1 convolution is used to assign each of the 64 component vectors to the two classes used (background and defect).

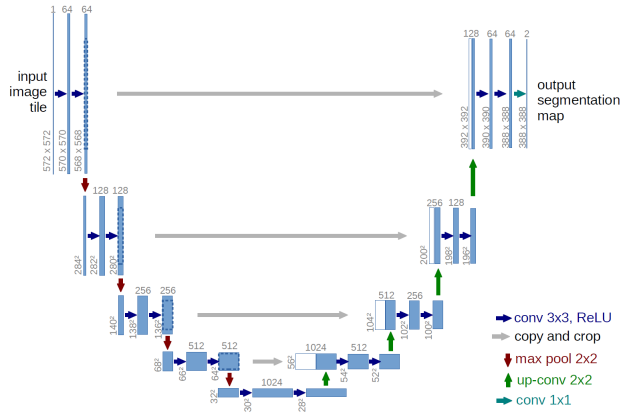


Fig. 2: U-Net architecture.

During the explanation of the architecture, we commented the terms "ReLU" and "Max pooling."

The rectified linear activation function, or **ReLU** for short, is indeed a function that, if the input is positive, outputs the input directly; else, it outputs zero.

Max Pooling is a pooling operation that calculates the maximum value for patches of a feature map and uses it to create a downsampled (pooled) feature map.

3.4 Patching

In order to increase batch size on the limited available hardware resources and to be able to keep the original image resolution, patching was implemented. Patching is a technique for dividing an image into smaller, fixed-size images.Besides reducing resource costs, patching can also improve the output of the model by increasing the variation of images during training. This is of especially great benefit considering the small size of the available dataset.

Patching is done in two ways in this project. A random patching method is used for training and validation. Random patching gets patches from an image at random positions. These positions will be different for each image.

The randomized patching results in more variation in the input images that the model can train on.

For testing, fixed patching is used. As the name suggests, fixed patching divides the image into four evenly sized patches. Fixed tiling is used here such that the testing image is processed fully and can be stitched back together at the end to generate a full result.

3.5 Hardware and Software

Table 1 shows the hardware specifications of the virtual machine used during this project.

Hardware specifications	
CPU	8 cores @ 2.4GHz
RAM	14GB
GPU Model	Nvidia GeForce RTX 2070 SUPER
GPU Memory	8GB
CUDA Version	11.0

Table 1: Hardware specifications of the virtual machine

The software used in this project is written in Python. In Python, the main packages used are PyTorch[13] and Albumentations[12].

3.6 Evaluation Metrics

The following metrics are used to determine the validity of a model's predictions.

3.6.1 Intersect over Union (IoU)

When evaluating the performance of a segmentation model, the intersect over union is one of the most commonly used metrics. This metric, also known as the Jaccard index, represents the area between the predicted segmentation and the ground truth. It is calculated like this:

$$IoU = \frac{AreaOfOverlap}{AreaOfUnion}. \quad (1)$$

The index ranges between 0 and 1, 0 representing no overlapping and 1 being a perfect match to ground truth.

3.6.2 Precision and recall

The precision is defined as the ratio of accurately predicted positive observations to the total predicted positive observations. In segmentation, this means the ratio of how many of the predicted pixels of a class fall within the ground truth of that class. It is calculated as such:

$$Precision = \frac{TP}{TP+FP}. \quad (2)$$

Recall is defined as the ratio of correctly predicted positive observations to the total possible observations as given by the ground truth. In segmentation, the metric defines the percentage of the ground truth that was detected, disregarding false positives. The recall is calculated as shown:

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

In these calculations TP stands for True Positives, FP stands for False Positives, and FN stands for False Negatives. Both metrics range from 0 to 1, where generally a higher score is better.

3.6.3 F1-Score

The F1-score is a weighted average of precision and recall metrics. It is a measure of a model's accuracy. Similar to the precision and recall, it ranges from 0 to 1, where higher is generally better. It is calculated as shown:

$$F1Score = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision} \quad (4)$$

4 Experiments and Results

In this section, experiments will be described and their results reported.

4.1 Experiments

Several experiments were run on the various datasets. They can be divided into the following categories; regular supervised image segmentation, transfer learning, and data augmentations (DA).

All experiments done are run four times, and the figures used in this paper are the average of those numbers. This is done to avoid extreme highs and lows due to randomized starting conditions.

The variations exposed do not include all the combinations of the parameters. The best combination is chosen by testing all of them on a single dataset, and then the transfer learning is just applied with the best parameters. You can see the combinations in the C in the 7

For each experiment, the above mentioned metrics of IoU, F1-Score, Precision, and Recall are noted. Experiments are run for 100 epochs, and it saves the best found model for testing. Learning rate, batch size, and number of tiles are varied between different experiments and will be mentioned per experiment.

4.1.1 Regular Supervised Image Segmentation

To determine whether a U-Net image segmenter can perform well, the first experiments that need to be performed are on the Ship Defect dataset without any transfer learning or data augmentations. This creates a baseline to compare further experiments to.

4.1.2 Transfer Learning

In order to improve the performance of the regular experimentation, we experimented with transfer learning. Transfer learning aims to provide a framework to utilize previously-acquired knowledge to solve new but similar problems more effectively by adding more data for a model to train on. Due to the similar nature between cracks in metal and cracks in concrete, the Concrete Break dataset was used as the basis for the transfer learning. The transfer learning was done in two different ways. One was to use the Concrete Break dataset for both training and validation, and only test on the Ship Defect dataset.

The other method was to combine the Concrete Breaks and the Ship Defect datasets for training and validating, and then test only on the Ship Defect images.

4.1.3 Varying Parameters

From the beginning, all the experiments were used the default parameters. So after the addition of the new datasets, it was necessary to start experimenting with some other possibilities.

The model contains 20 parameters, of which 8 are most suitable for modification. These next parameters were changed and combined to study the performance of the model in different environments.

These parameters involve all the possibilities of the program. With these parameters include tiling options like the number of tiles, tile size, and the batch size, giving control over how it splits the full images into small ones. There are also quantitative parameters that control how many cycles the model will work (epochs), how many start filters and depth the model will have, how many epochs can lead to over-fitting of the training dataset (patience), or how much to change the model in response to the estimated error each time the model weights are updated (learning rate).

There are also standard parameters that we use, like the 2 classes (defect and background), the 3 channels (RGB), or the workers (a generic term for application instance with a specific purpose). The basic parameters that let the model work, like the Adam optimizer [14], the way the loss is calculated (Binary Cross Entropy), or how we reduce the learning rate if there is no improvement using the ReduceLROnPlateau Scheduler [14].

Parameters	
Number of Tiles	**2**, 4
Tilling (High x Weight)	128 x 128, **256 x 256**
Patch Size	2, **3**, 4, 8
Learning Rate	0.002,**0.005**, 0.007
Epochs	**100**
Depths	3, **5**, 6
Start Filters	16, **64**
Patience	**20**
Up-mode	Transpose
Classes	2
Workers	8
Channels	3
Optimizer	Adam
Optimizer Learn Rate	0.005
Optimizer Weight Decay	0
Loss	Binary Cross Entropy
Loss Arg Max	True
Scheduler	ReduceLROnPlateau
Scheduler Patience	20

Table 2: Possible parameters of the model (Default parameters in bold)

4.1.4 Data Augmentation

In order to study the performance of the model in different environments, the next DA techniques were implemented:

Used Data Augmentation	
Grayscaling	
Probability	0.3 - 0.5 - 1
Random Contrast	
Limit	0.02 - 0.01
Probability	0.3 - 0.5 - 1
Rotation	
Degrees	45º-90º
Probability	0.5 - 1

Table 3: Augmentations used

Gray scaling is a DA technique that allows training and validation on images without color by turning the 3 RGB channels into a single one. Most of the images are very different in the dataset, so applying contrast and rotation can train the model on a greater variation of break directions and more distinctive contrasts. Each DA technique tries to improve the segmentation of defects in ships.

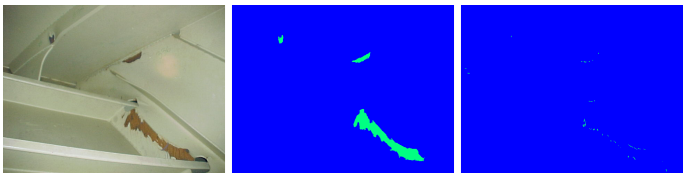
4.2 Results

As mentioned in section 4.1.1, first, a baseline has to be established. For this, all parameters are set to the default as seen in table 2.

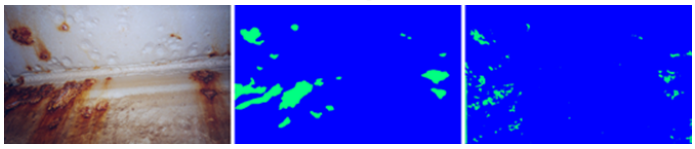
Base experiment results

Dataset	F1-Score	IoU	Precision	Recall
Ship Defects	0.1785	0.0995	0.5145	0.1345
MixedDataset3	0.192	0.137	0.4	0.172
MixedDataset1	0.0868	0.046	0.2635	0.0555
MixedDataset2	**0.246**	**0.14**	**0.386**	**0.282**
Concrete Breaks	0.142	0.076	0.295	0.921

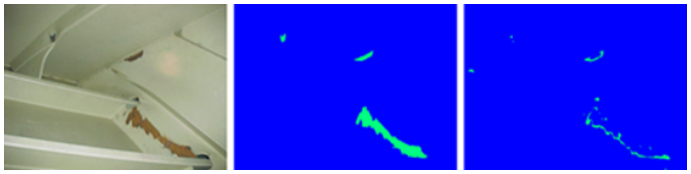
Table 4: Baseline Experiment Results (Default Parameters)



(a) Visual Results of the Ship Defects baseline test.



(b) Visual Results of the MixedDataset3 baseline test.



(c) Visual Results of the MixedDataset1 baseline test.

(d) Visual Results of the MixedDataset2 baseline test.

(e) Visual Results of the Concrete Breaks baseline test.

Fig. 3: Visual results of the Baseline Experiments performed

Looking at both table 4 and figures 3a and 3e, it appears that training on the Ship Defects dataset itself returns few but more precise detection of damage, having high precision but low recall, whereas when training on Concrete Breaks alone, the model over-detects a lot, resulting in high recall but low accuracy. The figures 3b, 3c, and 3d show a transition from high precision but low recall, to low recall but high precision. A sudden jump in detection between 3d and 3e shows that the Ship defects dataset, even with just a small number of images present, has a large impact on the pixel classification.

Plotting out the precision, recall and F1-Score against one another, see figure 4, shows that the precision becomes lower as the percentage of concrete breaks images in the dataset goes up, whereas the recall goes up. It also shows the F1-Score remain rather low, with either the recall or precision pulling the F1-Score down.

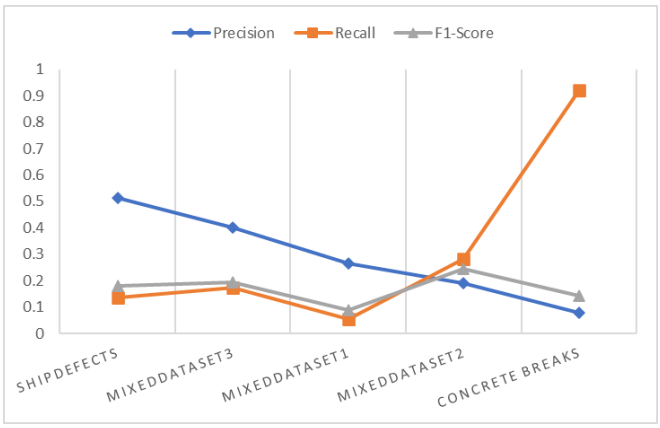


Fig. 4: Graph of the Precision, Recall and F1-Score per dataset.

4.2.1 Varying Parameters

You can find the result data of the various parameter changes and data augmentations in Appendix C.
Visual results are shown in Appendix D. The results indicate that the attempted domain shifting did not have a positive effect on model performance. The model performance with all tried data augmentations either returns similar results to the baseline experiments or worse results with both lower precision and recall. This can, for instance, be seen in table 8, where all results are either equal to, or worse than, the baseline results. Of note here is that the column with batch size 3 is the baseline result itself.

Table 7 shows the results of different parameters and data augmentations performed on the Concrete Breaks dataset. The results show roughly similar or worse performance with slight variations in precision or recall.

Other tables and figures in the mentioned appendices follow the same pattern of having similar or worse quantitative results, with the visual results agreeing.

5 Discussion and Conclusion

5.1 Discussion

At the start of the experiments, we realized that the limited size of the Ship Defect dataset hinders the ability to obtain reasonable segmentation predictions.

The proposed solution was to experiment with transfer learning and data augmentation. We used a dataset with images of concrete cracks to do some experiments. We realized that the transfer learning used improves the recall in exchange for a decrease in precision. The results of these experiments show that using the right dataset combination can improve or stabilize the values of recall, precision, and F1-score (respectively 0,189–0,2815–0,246).

The results show that modifying the batch size and the number of tiles barely improves the precision and recall. These models are still not good enough to detect anything at all.

Finally, the experimentation with the data augmentations provided some more results. The rotation didn't improve the performance of any value, such as the results in the gray scaling experiments, where the recall is high but the precision is nearly zero. In most of the data augmentation cases, there is no improvement, but in the experiment in the MixedDataset3 dataset with random contrast, it improves the performance, stabilizing the recall while decreasing the precision.

This project started with a clear limitation: the size of the used datasets. Even so, the experimentation proved that transfer learning methodology and a certain combination of datasets can improve the segmentation of the ship's damage. It is well known that with a set of 80 training images, it is difficult to have better results, but with more images, the performance of the U-Net model should improve properly.

Nowadays, there are other projects with a similar goal but in a different environment. Some other options to improve the results should be to find more similar datasets. For example, there are studies trying to detect the same damage but in bridges. As is shown in this paper, the combination of the right number of images from each dataset can improve the results.

5.2 Conclusion

To find the answer to the research question: *Can Artificial Neural Network based image segmentation be implemented to detect damage on the hull of ships?*, the sub-questions will need to be answered.

The first question: *Can a U-Net image segmentation network achieve good performance in segmenting damages on ships? Where good performance is based primarily on the precision and recall, and a score above 0.8 will be considered a good score.*
From table 4 it seems that U-Net can give a high precision but low recall when it comes to the Ship Defect dataset. This means that it misses a lot of the damage in each image, possibly missing sections entirely. Given the purpose is to detect damage, recall is the more valuable metric to focus on, with precision being secondary. With that as a basis, the transfer learning results performed better. As can be seen in figure 4, the recall increases with the percentage of Concrete Breaks images. It does, however, reduce precision. All in all, U-Net can be said to give decent results so long as the focus is on recall.

The second question, *Can a similar, but not directly related, dataset be used to improve model performance?*, is focused on transfer learning. As discussed in the previous section, with the use of the Concrete Breaks dataset, the recall can be increased to a high level. With a focus on recall rather than precision in segmentation, it can be said that an out of domain dataset has increased model performance.

For the last question, *Can data augmentation help improve model performance?*, the focus lay on data augmentation. As such, multiple experiments were performed with different augmentations. These augmentations were to focus on making the datasets look more similar, in essence shifting the out of domain Concrete Breaks dataset closer to the Ship Defect dataset. Here, however, the results were not positive. In most cases the data augmentations performed, as listed in table 3, the results were either similar or worse than their non-augmented equivalent. As such, data augmentation did not help model performance.

With these questions answered, an answer can now be given to the main research question. Simply said, yes, an Artificial Neural Network based semantic segmentation model can be used to segment damage on ships. There is, however, more to be done before it can be put to practical use. As it stands, the precision is too low for good detection. This paper serves more as a proof of concept here. It has been proven that a model can be trained to segment damage to ships. For practical use, more work should be performed to improve performance. A few options for such future work have been outlined below in the Future Work section.

5.3 Future Work

There are several options which might be able to improve upon the results provided in this paper. One such option would be to either increase the size of the datasets. Perhaps with more data for transfer learning can improve model performance. Better yet, would be to increase the size of the relevant dataset directly, as this would give the U-Net model more to train on.

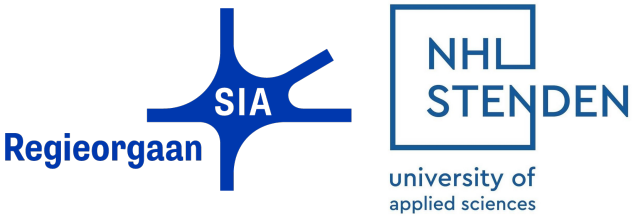
Perhaps alongside more data, increasing the quality of the data itself would be a good idea. Currently most data in the Ship Defect dataset is taken at angles and at times with poor lighting. Thus it is recommended that new images are taken from the front and with adequate lighting.

A different approach might be to use another segmentation model. This paper focused on U-Net for its excellent performance in medical image segmentation, but perhaps another model would be more successful with the limited data available.

Perhaps another approach entirely would be worthwhile. Considering the limited dataset this project started out with Few-Shot learning, as discussed in [15], could improve results further.

Acknowledgements

This project is financially supported by Regieorgaan SIA (part of NWO) under the RAAK PRO project Ship Defect



REFERENCES

[1] UNCTAD review of maritime transport. https://unctad.org/es/node/29022. Accessed: 2022-02-28.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[3] Yuan Yao, Yang Yang, Yanpeng Wang, and Xuefeng Zhao. Artificial intelligence-based hull structural plate corrosion damage detection and recognition using convolutional neural network. *Applied Ocean Research*, 90:101823, 2019.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[5] Brandon J. Perry, Yanlin Guo, and Hussam N. Mahmoud. Automated site-specific assessment of steel structures through integrating machine learning and fracture mechanics. *Automation in Construction*, 133:104022, 2022.

[6] Manali Shaha and Meenakshi Pawar. Transfer learning for image classification. In *2018 second international conference on electronics, communication and aerospace technology (ICECA)*, pages 656–660. IEEE, 2018.

[7] Annegreet Van Opbroek, M Arfan Ikram, Meike W Vernooij, and Marleen De Bruijne. Transfer learning improves supervised image segmentation across imaging protocols. *IEEE transactions on medical imaging*, 34(5):1018–1030, 2014.

[8] Thomas Koch, Sankaranarayanan Natarajan, Felix Bernhard, Alberto Ortiz, Francisco Bonnin-Pascual, Emilio Garcia-Fidalgo, and JJC Corcoles. Advances in automated ship structure inspection. In *International Conference on Computer Applications and Information Technology in the Maritime Industries*, 2016.

[9] Eric Loran Bianchi. *Data-driven Infrastructure Inspection*. PhD thesis, Virginia Tech, 2022.

[10] Randall Balestriero, Leon Bottou, and Yann LeCun. The effects of regularization and data augmentation are class dependent. *arXiv preprint arXiv:2204.03632*, 2022.

[11] Randall Balestriero, Ishan Misra, and Yann LeCun. A data-augmentation is worth a thousand samples: Exact quantification from analytical augmented sample moments. *arXiv preprint arXiv:2202.08325*, 2022.

[12] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020.

[13] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch.

[14] Jason Brownlee. Gentle introduction to the adam optimization algorithm for deep learning, Jan 2021.

[15] Athanasios Voulodimos, Eftychios Protopapadakis, Iason Katsamenis, Anastasios Doulamis, and Nikolaos Doulamis. A few-shot u-net deep learning model for covid-19 infected area segmentation in ct images. *Sensors*, 21(6), 2021.

A Software Libraries

Software library versions	
PyTorch	1.8.2
Albumentations	1.1.0

Table 5: Versions of the most used software libraries

B Datasets Information

B.1 Images of Ship Defects

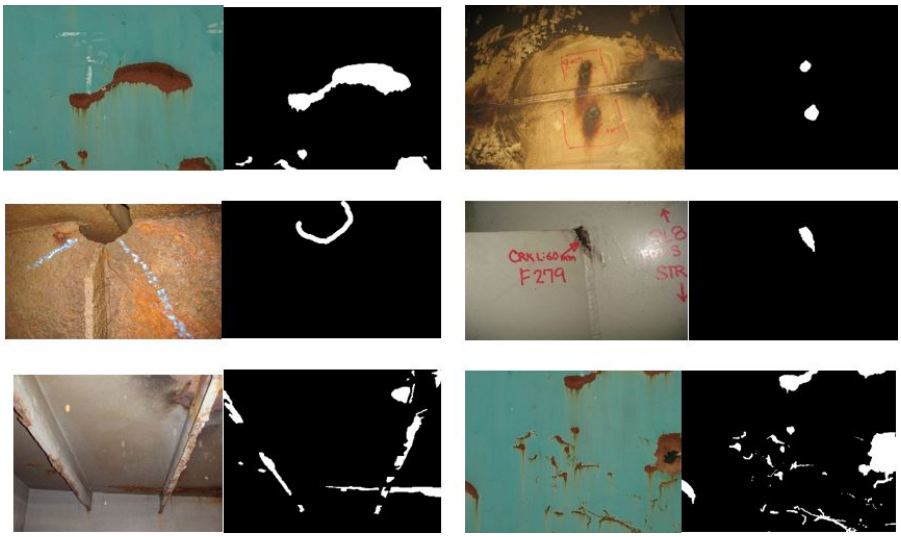


Fig. 5: Ship Defect Images

B.2 Images of Concrete Cracks

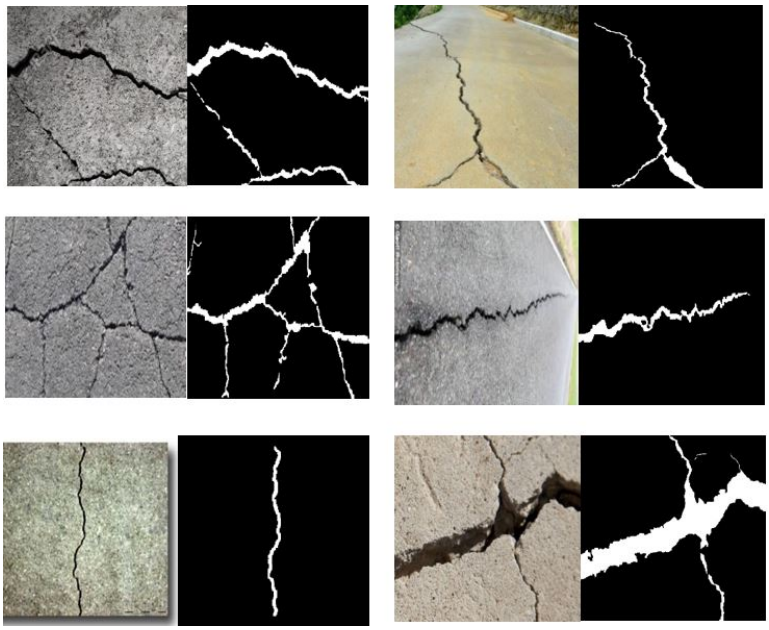


Fig. 6: Concrete Break Image

B.3 Proportions of the Created Datasets

Name	Portions		Training		Validation		Testing	
	Ship Defects	Breaks	Ship Defects	Breaks	Ship Defects	Breaks	Ship Defects	Breaks
MixedDataset1	50%	50%	40	40	15	15	10	0
MixedDataset2	33%	67%	20	40	8	15	10	0
MixedDataset3	67%	33%	40	20	15	8	10	0

Table 6: Proportions of the Mixed Dataset.

C RESULTS DATA

Testing of different parameters and data augmentation on the Concrete Breaks dataset results

Different Parameters	**F1-Score**	**IoU**	**Precision**	**Recall**
Default Parameters	0.146	0.076	0.0768	0.921
Depth = 3	0.144	0.0755	0.076	0.922
Depth = 3 Start Filters = 16	0.1385	0.076	0.0.837	0.875
Learning Rate = 0.002	0.1455	0.0785	0.0795	0.8735
Learning Rate = 0.007	0.141	0.076	0.077	0.900
Data Augmentation	Rotation			
Probabilty = 0.5 Rotation = 45º	0.144	0.078	0.079	0.90675
Probabilty = 1.0 Rotation = 45º	0.162	0.089	0.092	0.855
Probabilty = 0.5 Rotation = 90º	0.132	0.070	0.071	0.932
Probabilty = 1.0 Rotation = 90º	0.138	0.0745	0.07475	0.929
Data Augmentation	Random Contrast			
Probabilty = 0.5 Limit = 0.2	0.152	0.074	0.0755	0.907
Probabilty = 1.0 Limit = 0.2 = 45º	0.148	0.068	0.068	0.951
Probabilty = 0.5 Limit = 0.5	0.129	0.069	0.0695	0.947
Probabilty = 1.0 Limit = 0.5	0.150	0.0813	0.083	0.897
Data Augmentation	Grayscaling			
Probabilty = 0.2	0.081	0.057	0.057	0.961
Probabilty = 0.5	0.106	0.056	0.056	0.942
Probabilty = 0.7	0.131	0.070	0.071	0.940
Probabilty = 1.0	0.112	0.0653	0.066	0.930

Table 7: Tested Parameters (Best parameters in **bold)**

Number of Tiles and Batch size augmentation results

Batch Size	2		3		4		8	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Ship Defects	0,263	0,15	0.515	0.134	0,255	0,017	0,334	0,018
MixedDataset1	0.211	0.019	0,264	0,056	0,345	0,02	0,303	0,145
MixedDataset2	0,185	0,032	0,189	0,282	0,187	0,152	0,160	0,084
MixedDataset3	0,285	0,0272	0,255	0,017	0,241	0,099	0,344	0,0148
Concrete Breaks	0,076	0,886	0,077	0,921	0,085	0,898	0,069	0,916

Table 8: Batch size variation Results (Number of Tiles = 2)

Number of Tiles Batch size augmentation results

Batch size	2		3		4	
	Precision	Recall	Precision	Recall	Precision	Recall
Ship Defects	0,288	0,093	0,224	0,095	0,334	0,062
MixedDataset1	0,317	0,14	0,36	0,014	0.22	0.015
MixedDataset2	0,119	0,119	0,308	0,027	0,378	0,047
MixedDataset3	0,1815	0,02175	0,11875	0,1185	0,21925	0,166
Concrete Breaks	0,083	0,823	0,074	0,916	0,087	0,895

Table 9: Batch size variation Results (Number of Tiles = 4)

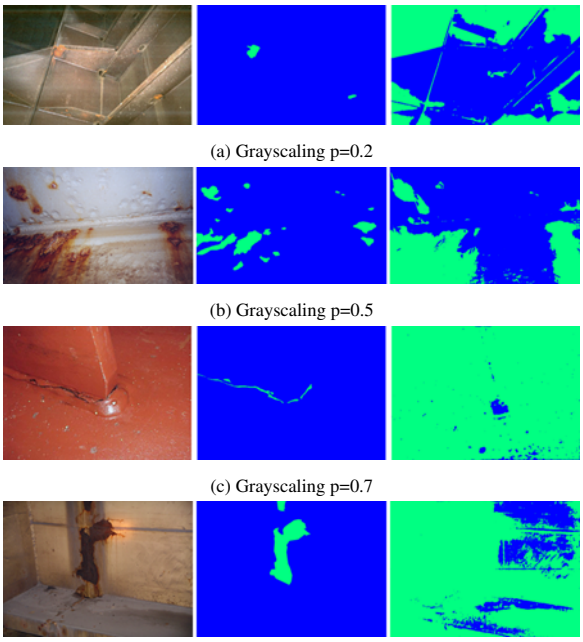
Random Contrast results				
Dataset	**F1-Score**	**IoU**	**Precision**	**Recall**
Ship Defects	0.126	0.069	0.331	0.086
MixedDataset1	0.054	0.029	0.231	0.033
MixedDataset2	0.087	0.063	0.119	0.113
MixedDataset3	0.032	0.016	0.249	0.017
Concrete Breaks	0.139	0.074	0.076	0.907

Table 10: Random Contrast Results (Probability= 0.5 Limit= 0.2)

Rotation results				
Dataset	**F1-Score**	**IoU**	**Precision**	**Recall**
Ship Defects	0.054	0.028	0.231	0.03
MixedDataset1	0.077	0.04	0.249	0.046
MixedDataset2	0.1445	0.08	0.179	0.147
MixedDataset3	0.104	0.059	0.477	0.068
Concrete Breaks	0.162	0.089	0.092	0.855

Table 11: Rotation Results (Probability= 1 Rotation= 45º)

D Visual Results



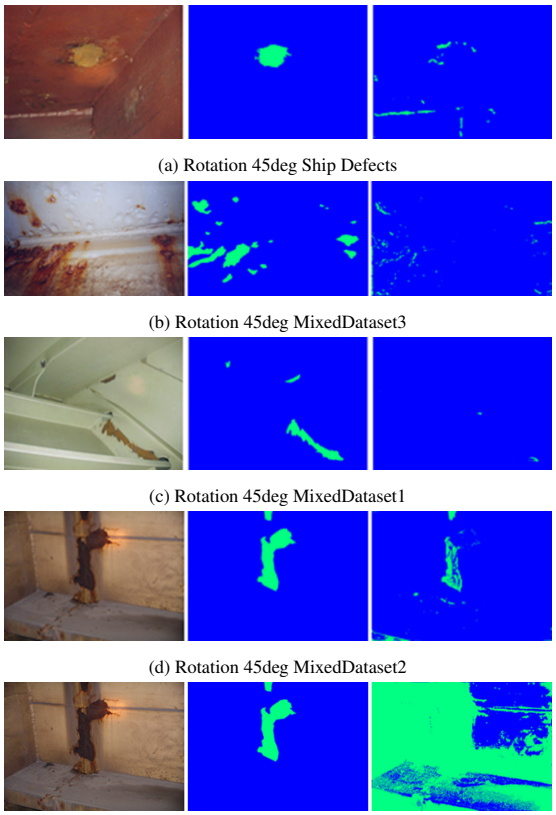
(a) Grayscaling p=0.2

(b) Grayscaling p=0.5

(c) Grayscaling p=0.7

(d) Grayscaling p=1

Fig. 7: Grayscaling data augmentation visual results



(a) Rotation 45deg Ship Defects

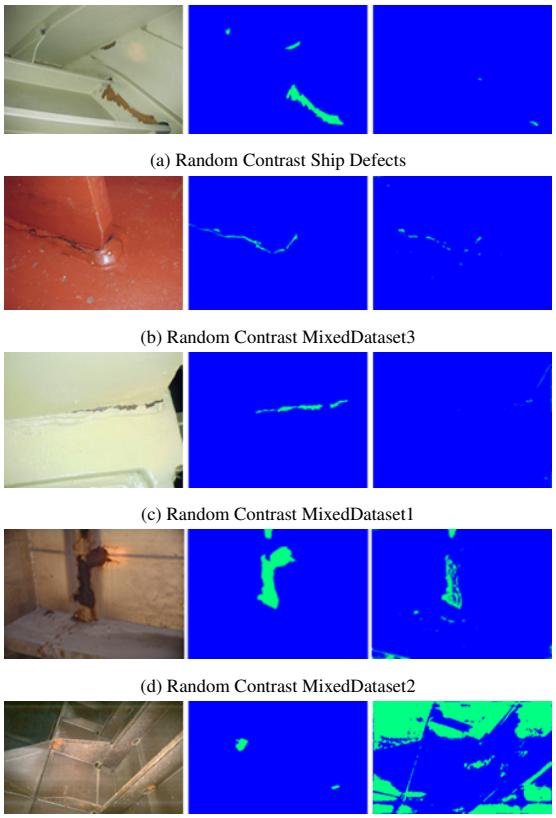
(b) Rotation 45deg MixedDataset3

(c) Rotation 45deg MixedDataset1

(d) Rotation 45deg MixedDataset2

(e) Rotation 45deg Concrete Breaks

Fig. 8: Rotation 45 degrees p=1 data augmentation visual results



(a) Random Contrast Ship Defects

(b) Random Contrast MixedDataset3

(c) Random Contrast MixedDataset1

(d) Random Contrast MixedDataset2

(e) Random Contrast Concrete Breaks

Fig. 9: Random Contrast limit=0.2 p=0.5 data augmentation visual results