

The background of the cover is a photograph of the front of a white car, likely a Toyota Camry, with a license plate that reads 'CL56341'. A blue vertical bar is positioned on the left side of the image. A black rectangular box is overlaid on the top half of the image, containing the title and author's name in white and blue text.

Generative Adversarial Networks for Shadow Removal to Improve Semantic Segmentation for Au- tonomous Driving

Daan Bruggink

Master of Science Thesis

Generative Adversarial Networks for Shadow Removal to Improve Semantic Segmentation for Autonomous Driving

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Systems and Control at Delft
University of Technology

Daan Bruggink

February 2, 2023

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of
Technology

The Netherlands Organisation for Applied Scientific Research (TNO) has supported the research conducted in this thesis. Their help is gratefully acknowledged.
The front cover image is adapted from the Waymo website: Waymo LLC [1]



Copyright © Delft Center for Systems and Control (DCSC)
All rights reserved.

Abstract

Traversability estimation is a key component in autonomous driving tasks. In many applications, semantic segmentation is used to pixel-wise classify a visual scene. The pixel-wise segmented map is used to estimate the traversability of different environments. The semantic segmentation accuracy can drop if environmental conditions change. The introduction of shadow in images can cause the segmentation network to misclassify pixels, which leads to an inaccurate traversability estimation. This inaccurate estimation could lead an autonomous vehicle to deviate from traversable paths, leaving it unable to continue or even cause accidents.

To increase the segmentation accuracy in shadow conditions, shadow removal before semantic segmentation is proposed. In this research, the supervised Dual Hierarchical Aggregation Network (DHAN) and unsupervised cycle-based Shadow Generative Adversarial Network (SGAN) are used for shadow removal before semantic segmentation with Segmentation Network (SegNet). The shadow removal networks are evaluated on two datasets, containing image triplets, consisting of shadow, shadow-free and shadow-mask images. The structural similarity is calculated for complete images and the non-shadow regions by inverting the shadow-masks. The networks are applied to the Cambridge-driving Labeled Video Database (CamVid) dataset to evaluate the change in segmentation accuracy. In a second set of experiments, the DHAN shadow removal network is retrained on multiple datasets containing synthetic shadows. The obtained DHAN networks are tested on shadows with increasing intensity. The retrained DHAN networks are compared to evaluate the training dataset for shadow removal to increase segmentation accuracy.

The experiments show that the DHAN increases the structural similarity of 99.8% and the SGAN for 74.5% of the datasets. After retraining on the synthetic shadow dataset, segmentation accuracy after DHAN shadow removal increases the segmentation Pixel Accuracy (PA) for a maximum of 91% and the segmentation mean Intersection over Union (mIoU) for a maximum of 88% of the images of the CamVid dataset. We conclude that segmentation accuracy increases after DHAN shadow removal if the DHAN is trained on the synthetic shadow dataset.

Table of Contents

Acknowledgements	xiii
1 Introduction	1
1-1 Research questions	2
1-2 Thesis contribution	3
1-3 Outline	3
2 Theoretical background	5
2-1 Neural Networks	5
2-1-1 The perceptron	5
2-1-2 Multilayer perceptron	6
2-1-3 Activation functions	8
2-1-4 Function approximation using Neural Networks	9
2-2 Parameter estimation	10
2-2-1 Loss function	10
2-2-2 Gradient descent	10
2-2-3 Backpropagation	11
2-3 Specialized Neural Networks	12
2-3-1 Convolutional Neural Network	12
2-3-2 Encoder-decoder network structure	15
2-3-3 Generative Adversarial Networks	17
2-4 Bias, variance, overfitting and underfitting in Neural Networks	18
3 Related works	21
3-1 Semantic segmentation	21
3-1-1 Segmentation architecture	21
3-2 Dual Hierarchical Aggregation Network	24
3-2-1 Architecture	24

3-2-2	Dilated convolution	25
3-2-3	Attentive aggregation nodes	26
3-2-4	Spatial pooling pyramid	27
3-2-5	Loss function	27
3-2-6	Datasets	28
3-3	Shadow Generative Adversarial Network	29
3-3-1	Dual generator and discriminator network	29
3-3-2	Loss function	30
4	Methodology	33
4-1	Training setup	33
4-2	Evaluation metrics	34
4-2-1	Structural Similarity Index Metric	35
4-2-2	Pixel Accuracy	36
4-2-3	Intersection over Union	36
4-3	Experiments	37
4-3-1	Comparison of removal methods	37
4-3-2	Segmentation experiments	38
5	Results	41
5-1	Shadow removal methods	41
5-1-1	Structural similarity comparison	42
5-1-2	Visual performance and error maps	46
5-1-3	Image comparison in non-shadow regions	48
5-2	Segmentation performance	53
5-2-1	Improvement ratio analysis	53
5-2-2	Visual inspection	55
5-3	Shadow analysis	55
5-4	Results on synthetic shadows	59
5-4-1	Structural similarity comparison	59
5-4-2	Segmentation performance	61
6	Conclusion	65
6-1	Summary of results	65
6-2	Conclusion	66
6-3	Future work	68
A	Additional figures	71
A-1	Activation function supplement	71
A-2	ADAM learning rate scheduler	72

B Complimentary tables and figures	73
B-1 Complimentary tables	73
B-2 Complimentary figures	74
Bibliography	77
Glossary	83
List of Acronyms	83

List of Figures

2-1	The perceptron. The perceptron consists of n inputs $x_1, x_2, \dots, x_{n-1}, x_n$, weights w_1, \dots, w_{n-1}, w_n and the bias term w_0 . The inputs are multiplied with their respective weights and summed. The activation function σ uses this sum as input; its output is the final output of the perceptron, y . Adapted from: [2].	6
2-2	Depiction of a Multilayer Perceptron. The network has 1 input layer, 2 hidden layers and 1 output layer. The size of the input layer is n_0 , hidden layer 1 and 2 have sizes n_1 , and n_2 , respectively, and the output layer has size n_3	7
2-3	Schematic overview of a Convolutional Neural Network. An input image is convolved into several feature maps. The feature maps are then pooled for downsizing and feature selection. This process is repeated until the desired feature space is calculated. After the determination of the feature mapping, the feature map is passed through several fully connected layers to predict what animal is presented to the network. The output is a probability for each possible animal class (Dog, Cat and Deer). Adapted from: [3]	13
2-4	Convolutional operation on I with kernel K with stride $s = 1$. The kernel (blue) is moved over the image I , starting in the left top corner. By moving the kernel one pixel to the right (stride=1) while calculating at each intermediate step, we get the convolutional mapping $I * K$. From: [4]	13
2-5	Pooling of a 4×4 matrix with pool size 2×2 . Adapted from: [5]	14
2-6	Example of an encoder-decoder network structure. The encoder utilises a set of pooling and convolutional layers to generate the lower dimensional latent space of the input image. The decoder takes the latent space as input and generates a reconstruction of the input image X . Adapted from: [6]	15
2-7	Switch variable unpooling of a pooled image with a 2×2 window. The stored indices are used to place an element at its original index before the pooling step. From: [7]	16
2-8	Bed-of-Nails unpooling. Zero values are placed between the elements of a 2D feature space with spacing $p_b = 1$. From: [8]	16
2-9	Example of the transposed convolution step with stride $s = 1$ and padding $p = 0$. Each element of the input is multiplied with each element of the kernel and stored. In blue, the example of the multiplication of the first input element with the kernel is shown. The outputs of all the multiplications are summed. From: [9]	16

2-10	Structure of a Generative Adversarial Network. A noise vector is fed to the generator, that creates a fake image from this noise vector. The discriminator predicts if a fed image is "fake" or "real". From: [10]	17
2-11	Generative Adversarial Network trained simultaneously: we can see the discriminative distribution (blue dotted line) the data generating distribution (black dotted line) p_x and the generative distribution (green line) $p_g(G)$. The lower horizontal line is the domain from which z is sampled. The upper horizontal line is part of the domain of x . The arrows between the lower and upper horizontal lines show how the mapping $x = G(z)$ imposes the non uniform distribution p_g on the transformed samples. From: [11]	18
2-12	The relation between bias, variance, underfitting and overfitting. A high bias means underfitting on the training data, while high variance shows overfitting on the training data. A model that shows a good fit to the training data, has a balanced bias and variance. This is called the Bias-Variance Trade-Off. Adapted from: [12]	19
3-1	The SegNet Neural Network (NN) architecture. The SegNet architecture is based on an encoder-decoder structure. An input image is encoded using convolutional and max pooling layers with switch variables, storing the pooling indices. The resulting latent space is upsampled using switch variable unpooling and convolutional layers. The last layer is a soft-max classifier layer of the same size as the RGB input image, assigning each pixel to a class and generating a pixel-wise segmented image. From [13].	22
3-2	An example of the images in the CamVid dataset. The top row shows the scenery images taken with the camera. The bottom row shows the Ground Truth segmented images paired with the top row. The legend shows the twelve classes of the CamVid dataset.	24
3-3	Structure of the Dual Hierarchical Aggregation Network. A shadow image is encoded using the VGG16 network (1). The encoded features are passed through two convolutional layers of convolution size 1x1 and processed further by several dilated convolution layers and attentive aggregation nodes (2). The final two attentive aggregation nodes' results are processed by a convolutional layer, spatial pooling pyramid (4), and convolutional layer. The final outputs are a shadow-free image and its respective shadow mask. Adapted from: [14].	25
3-4	Schematic overview of dilated convolution. (a) Shows dilated convolution for $d = 1$, which is the same as normal convolution. (b) Shows dilated convolution with dilation rate $d = 2$. The kernel is moved over the input with spacing $d - 1$, increasing the spacing between the matrix convoluted with kernel K . Adapted from: [15]	26
3-5	Attentive Aggregation Node in the DHAN. The features of several layers or nodes are concatenated. This concatenated feature space is then pooled and passed through three fully connected linear layers, layers one and two containing linear (identity) activation functions and the final layer having a sigmoid activation function. The output of this sigmoid layer is multiplied by the original concatenated features, convoluted, and passed through another fully connected layer with sigmoid activation function. From: [14]	26
3-6	Overview of Spatial Pooling Pyramid (SPP). The feature maps are max pooled with three different window sizes. The first window is size $a \times b$, generating a 1x1 max pool matrix (gray). The second window is size $\frac{a}{2} \times \frac{b}{2}$, generating a 2x2 max pool matrix (green). The last window is size $\frac{a}{4} \times \frac{b}{4}$, generating a 4x4 max pool matrix (blue). The max pool values are stored as a vector, generating a fixed-length representation of the feature maps of size $21 \times d$. From [16]	28

3-7	(a) Example triplets from the ISTD dataset and (b) example pairs from the SRD dataset.	29
3-8	A dual generator-discriminator network with cycle consistency. In (a) , the generators G and F are shown, where $G : X \rightarrow Y$ and $F : Y \rightarrow X$ and their accompanying discriminators D_X and D_Y respectively. D_Y tries to distinguish fake images generated by G from images in dataset Y and D_X tries to distinguish fake images generated by F from images in dataset X . In (b) , the cycle consistency criterion is shown. The intuition is that an image mapped from X to Y by G , F should be able to map it back into its original domain. This is called the forward cycle consistency. In (c) , we see the same as in (b) , but for mappings from Y to X . This is called the backward cycle consistency. From: [17]	30
3-9	The SGAN loss pairs. In (a) we see the shadow cycle consistency loss (yellow), the shadow identity loss and adversarial (discriminator) loss of the images generated by G_f (skyblue). In (b) , the shadow-free cycle consistency loss (yellow), shadow adversarial (discriminator) loss (skyblue) and shadow-free identity loss (green) are shown. From: [18]	31
5-1	Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the (a) ISTD and (b) SRD datasets. For all three removal methods, the structural similarity is shown in grayscale (gray violins) and for each of the three color channels (red-, green-, and blue violins).	43
5-2	Structural similarity improvement ratio between Input and DHAN - and SGAN for the (a) ISTD and (b) SRD datasets. For both networks, the structural similarity in grayscale (gray violins) and for each of the three color channels (red-, green-, and blue violins) is shown.	44
5-3	The correlation between the masksize and structural similarities of the Input, DHAN and SGAN. The top row shows the results on the ISTD dataset. The bottom row shows the results on the SRD dataset.	46
5-4	Best DHAN and SGAN (DHAN ⁺ , SGAN ⁺) and second worst DHAN (DHAN ⁻) and worst (SGAN ⁻) performances on the ISTD dataset. Input shows the original image containing a shadow to be removed. Mask contains the binary (black and white) mask of the shadow to be removed. The Ground Truth (GT) is the target shadow-free image. DHAN and SGAN show the results of the DHAN and SGAN networks on the Input. The Shadow SSIM map, DHAN SSIM map, and SGAN SSIM map show the structural similarity per pixel for the input, DHAN output, and SGAN output with respect to the Ground Truth. The maps show the structural similarity range from black (structural similarity = 0) to white (structural similarity = 1).	47
5-5	The best (+) DHAN and SGAN, second worst DHAN (DHAN ⁻) and worst (SGAN ⁻) performances on the SRD dataset. Input shows the original image containing a shadow to be removed. Mask contains the binary (black and white) mask of the shadow to be removed. The GT is the target shadow-free image. DHAN and SGAN show the results of the DHAN and SGAN networks on the Input. The shadow SSIM map, DHAN SSIM map, and SGAN SSIM map show the structural similarity per pixel for the input, DHAN output, and SGAN output with respect to the Ground Truth. The maps show the structural similarity range from black (structural similarity = 0) to white (structural similarity = 1).	47
5-6	SSIM(I'_s, I'_f) for the ISTD (a) and SRD (b) datasets. The SSIM metric is split into its three components; the (a) Luminance, (b) Contrast and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.	49

5-7	SSIM(I'_{DHAN}, I'_f) for the (a) ISTD and (b) SRD datasets. The SSIM metric is split into three components; (a) Luminance, (b) Contrast, and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.	50
5-8	SSIM(I'_{SGAN}, I'_s) for the (a) ISTD and (b) SRD datasets. The SSIM metric is split into three components; (a) Luminance, (b) Contrast, and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.	51
5-9	Input SSIM(I'_s, I'_f) plotted against SSIM(I'_{DHAN}, I'_f) and SSIM(I'_{SGAN}, I'_f) for the ISTD (top row) and SRD (bottom row) datasets.	52
5-10	Correlation between structural similarity and masksize	52
5-11	Improvement ratios of the (a) PA and (b) mIoU on the CamVid dataset. The improvement ratios are shown for the DHAN and SGAN network trained on the ISTD and SRD datasets.	54
5-12	The 50'th, 100'th, 150'th and 200'th of the CamVid dataset images I_{Cam} . The post shadow removal (a) DHAN and (b) images are depicted, including the difference map between the shadow removed and the original CamVid image. The intensity difference maps are shown in grayscale.	56
5-13	width =	57
5-14	The 50'th, 100'th, 150'th and 200'th of ISTD and SRD datasets respectively. The shadow image is subtracted from the shadow-free image to find the intensity difference.	58
5-15	Pixel intensity difference I_{diff} distribution for the ISTD and SRD dataset. The pixel intensity difference is discrete and can only be an integer value.	58
5-16	Structural similarity improvement ratio of each of the trained DHAN _{<i>i</i>} networks on the DataSyn ₁₁ dataset.	60
5-17	Structural similarity improvement ratio of each of the trained DHAN _{<i>i</i>} networks on the DataSyn ₁ and DataSyn ₁₀ datasets.	60
5-18	Improvement ratios of the (a) PA and (b) mIoU. The ratios have been calculated according to Equations 5-2 and 5-3. The improvement ratio is shown for each retrained DHAN _{<i>i</i>} network.	62
5-19	Pixel Accuracy and Intersection over Union segmentation accuracy on the DataSyn _{<i>i</i>} datasets before shadow removal.	64
A-1	More complete overview of activation functions and their graphs. This figure states the activation function, its equation, its use in Neural Networks and its visualisation in 1D graph form. From: [19].	71
B-1	Visualisation of increasing shadow intensities on CamVid image, where the shadow intensity increases from from left to right, top to bottom.	74
B-2	Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the (a) ISTD and (b) SRD datasets where images SSIM(I'_s, I'_f) < 0.90 are removed. For all three removal methods, the structural similarity is shown in grayscale (gray violins) and for each of the three color channels (red, green, and blue violins).	75

List of Tables

B-1	Complimentary table to Figure 5-16. The structural similarity improvement ratio median, quartile, mean, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_{11}$ dataset. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.	73
B-2	Complimentary table to Figure 5-17. The structural similarity improvement ratio median, mean, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_1$ ($i = 1$) and $DataSyn_{10}$ ($i = 10$) datasets. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.	73
B-3	Complimentary table to Figure 5-18a. The PA improvement ratio median, mean, minimum, maximum, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_{11}$ dataset. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.	74
B-4	Complimentary table to Figure 5-18b. The mIoU improvement ratio median, mean, minimum, maximum, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_{11}$ dataset. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.	74
B-5	Complimentary table to Figure 5-19a. The median, mean, minimum, and maximum PA for the $DataSyn_i$ datasets without shadow removal. The correlation between $DataSyn_1$ to $DataSyn_{10}$ PA values and the shadow intensity is shown in the eleventh column.	74
B-6	Complimentary table to Figure 5-19b. The median, mean, minimum, and maximum mIoU for the $DataSyn_i$ datasets without shadow removal. The correlation between $DataSyn_1$ to $DataSyn_{10}$ mIoU values and the shadow intensity is shown in the eleventh column.	74

Acknowledgements

I would like to thank Pieter Piscaer and Frank ter Haar, who have helped me throughout the research presented in this thesis. The brainstorm sessions at TNO, the department of Intelligent Imaging, have not only generated more insight, but also helped me pass difficult moments. Second, I would like to thank Carlas Smith for his guidance, feedback and kind words in difficult times.

I would like to thank Dylan Kalisvaart twice. His guidance and feedback pushed me to maximise the results of this thesis. Furthermore, his never ending enthusiasm and always positive attitude have motivated me when I did not see the path forward.

Finally, my gratitude to all who have supported me throughout my studies. My friends and girlfriend, who were always there when I needed them, and my family, whose feedback and support never faltered.

Thank you all for an amazing end to this part of life!

Delft, University of Technology
February 2, 2023

Daan Bruggink

Chapter 1

Introduction

The development of autonomous transportation brings possible benefits to society; autonomous vehicles can prevent accidents caused by human error, decrease carbon emissions and increase travel comfort [20, 21, 22]. In spring 2022, an autonomous car picked up an engineer from his home, stopped for a coffee at the local Starbucks, and continued the journey to his workplace [23]. This self-driving vehicle, developed by Waymo, is one of the first fully autonomous cars to pick up a passenger, make an extra stop, and drop off its passenger without assistance from a driver. This example showcases the recent interest and development of autonomous vehicles.

Implementing autonomous vehicles into society to reap the benefits requires information about the surroundings. In known environments, such as cities, one of the key elements in solving autonomous driving challenges is localization and traversability estimation [24, 25]. The goal is to find where the vehicle is located in the environment (localization) and distinguish between obstacles, such as traffic and pedestrians, and traversable paths, such as roads (traversability estimation). Image-based vision methods such as object detection and segmentation are able to gain the understanding of the environment to perform traversability estimation [26, 27] and are the basis of short- and long-term path planning to assist in autonomous navigation around obstacles and stay on the road [28].

The increased interest in autonomous vehicles has not been limited to the public domain. Military and defense industries have been looking for ways to incorporate Unmanned Ground Vehicles (UGVs) into their arsenal to perform reconnaissance and target acquisition missions. UGVs can drastically reduce human casualties by increasing combat effectiveness and keeping human operators out of dangerous situations and environments. The Defence Advanced Research Projects Agency (DARPA) has announced the Robotic Autonomy in Complex Environments with Resiliency (RACER) program that seeks to find algorithms and Deep Learning (DL) approaches, a subfield of machine learning that uses data to build a model, that can replace the need for a human driver in military operations [29]. In contrast to autonomous vehicles for civil environments, the military UGVs need to navigate through harsh and often unknown terrain. The structured form of a city with roads, traffic signs, and buildings is

replaced with more diverse obstacles, such as boulders and crevices, without clear pathways around them [30]. To navigate through unknown environments, three control levels are considered: **(a)** Global path planning to find the optimal route from starting point to destination, **(b)** local navigation for obstacle avoidance, and **c** perception to understand the surroundings of the vehicle and its traversability [31]. In this thesis, the focus is on the perception task for traversability estimation.

A common method for traversability estimation based on a camera feed is semantic segmentation. Semantic segmentation is a DL method that assigns a class label, for instance road, grass or pedestrian, to each pixel in an image [27]. This method relies on a Convolutional Neural Network (CNN) that specializes in extracting information from high dimensional data that has a grid-like topology [32, 13]. The scene understanding properties of the semantic segmentation approach, where the classes are pre-defined based on the application, makes it a common choice in a great variety of vision-based applications. These applications include autonomous driving [9], robotics [33], medical imaging [34] and video surveillance [35].

However, current semantic segmentation approaches suffer from changing weather conditions such as rain, sun glare, and shadow, even causing crashes in a UGV competition in Korea [36, 37, 38, 39]. Shadows will occur in all environments depending on the time of day. Autonomous vehicles should be able to operate in all conditions, where cast shadows by the environment should not affect the traversability estimation. Since shadow removal before segmentation and its possible benefits have not yet been thoroughly researched, this thesis will focus on shadow removal to improve semantic segmentation for UGVs.

1-1 Research questions

The situation described in the previous section leads to the question whether shadow removal can benefit semantic segmentation. If shadow conditions can be removed from an image before segmentation, the accuracy of segmentation will likely increase. To fully comprehend this possible accuracy increase, this research tries to answer the following main research question:

"How can shadow removal improve semantic segmentation accuracy for traversability estimation?"

To formulate an answer to the main research question, two sub-questions are formulated. Answering these questions will provide additional information necessary to answer the main research question. The first sub-question is formulated as follows:

"1. How can shadow be removed without changing pixel values in non-shadow regions of an image?"

The proposed method for shadow removal before segmentation needs to be evaluated on both the shadow removal and segmentation sides. We define the second sub-question:

"2. How can the effect of shadow removal on semantic segmentation be compared to semantic segmentation without shadow removal?"

1-2 Thesis contribution

In this thesis, we compare semantic segmentation performance before and after shadow removal. In the first set of experiments, the supervised Dual Hierarchical Aggregation Network (DHAN) and unsupervised Shadow Generative Adversarial Network (SGAN), two algorithms that used for shadow removal, are compared to decide on a shadow removal approach. With supervised learning, datasets X and Y consisting of data pairs $\{x_i, y_i\}$ are available for training and testing the network, with x_i a shadow-free image and y_i the same image containing shadow. With unsupervised learning, datasets X and Y are not paired, and each dataset has specific characteristics, X containing shadow images and Y shadow-free images. We find the benefits and drawbacks of each method and compare the results on two datasets. The shadow removal methods are applied to the CamVid dataset, and the results are segmented using the Segmentation Network (SegNet).

In the second set of experiments, a synthetic shadow dataset is generated. The synthetic shadows are generated based on an evaluation of existing datasets containing shadow and shadow-free image pairs. The synthetic dataset is used to retrain the DHAN shadow removal method. The segmentation performance of SegNet is again evaluated after shadow removal on the CamVid dataset. With this study and its results, we offer insight into shadow removal as the first step in the semantic segmentation process used as a traversability estimator for autonomous vehicles. The findings show that the DHAN shadow removal approach is able to increase the structural similarity for almost all images. Furthermore, we see the effect of shadow removal before semantic segmentation on the segmentation accuracy.

1-3 Outline

This thesis contains six chapters and two appendices. In Chapter 2, the theory behind Neural Networks (NNs) is discussed to build a foundation for a better understanding of the networks used during this thesis. We also discuss the optimization technique for NNs and specialized networks, such as the CNN and Generative Adversarial Network (GAN). In the next chapter, Chapter 3, the three networks evaluated for shadow removal and semantic segmentation are presented. Additionally, the training strategy and loss function for each network is explained. The experiments conducted to evaluate shadow removal for segmentation are discussed in Chapter 4. Furthermore, the training parameters for each of the networks and the metrics for the evaluation of the results are stated. In Chapter 5, theory and methodology have been applied, and the results of the experiments are shown. The final chapter, chapter 6 concludes this thesis with a summary of the results and answers to the research questions. Furthermore, recommendations to improve this research and explore future research options are made.

Theoretical background

In this chapter, the process to create a Neural Network (NN) from its elementary building-blocks will be discussed. In Section 2-1, the elementary buildingblock of NNs, called the perception, is explained. The perception is used to create a Multilayer Perceptron Network (MLP) and the ability of a MLP to approximate a function is discussed. In Section 2-2, we start by explaining the parameter estimation of a NN. Then, the gradient descent and backpropagation algorithms to optimize the parameters of the network are stated. In Section 2-3, the Convolutional Neural Network (CNN) and Generative Adversarial Network (GAN) are discussed and finally, in Section 2-4, Machine Learning (ML) terminology is explained.

2-1 Neural Networks

NNs are a type of machine learning models that have the ability to learn from large and complex datasets. The idea of NNs is based on the same principle as the neurons in the human brain. It is composed of layers of interconnected neurons. An input signal is given to a neuron, processed by it, and an output signal is given. A set of neurons can then accomplish a task by working together. In 1990, one of the first NNs was used for classification [40]. Since then, NNs have come a long way, with MLPs, also called a Feedforward Neural Networks (FNNs), and specialized networks such as the CNNs and GANs as results.

2-1-1 The perceptron

The perceptron is the basis of NNs. A perceptron is a linear combination of n inputs $\mathbf{x} = [x_1, x_2, \dots, x_{n-1}, x_n] \in \mathbb{R}^n$, their respective weights $\mathbf{w} = [w_1, \dots, w_{n-1}, w_n]$, the bias term w_0 and an activation function σ . First, the summed inputs, weights and bias can be formulated as [2, 41]:

$$z = w_0 + \sum_{i=1}^n w_i x_i = w_0 + \mathbf{w}^T \mathbf{x} \quad (2-1)$$

where z is the linear combination of the inputs and their weights plus the bias term, w_0 . The output of the perceptron is based on z and activation function σ :

$$y = \sigma(z) \quad (2-2)$$

A complete overview of the perceptron is shown in Figure 2-1.

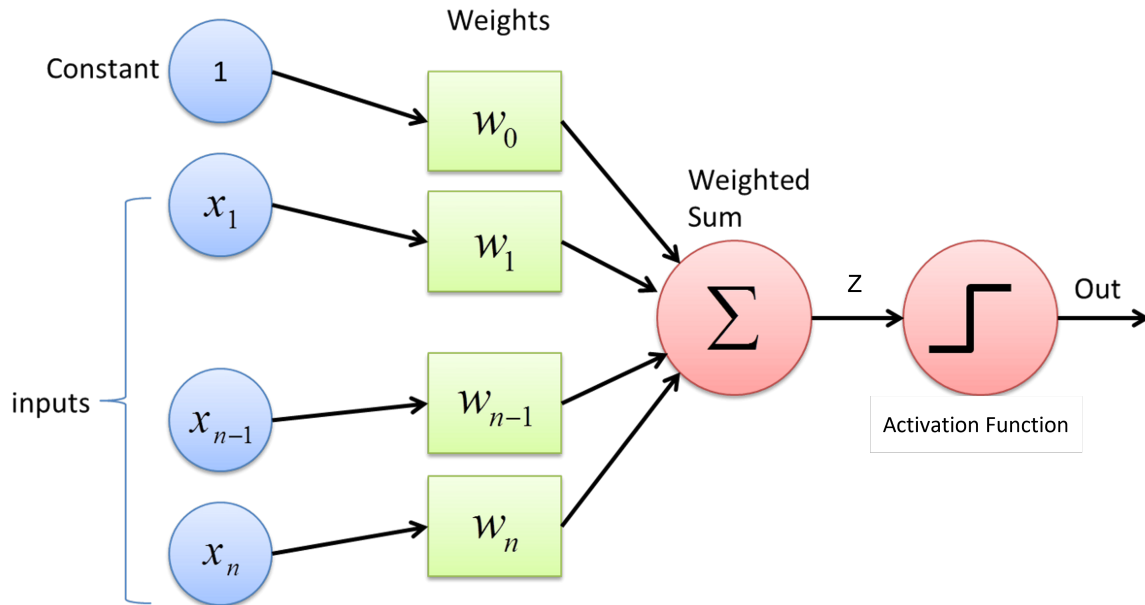


Figure 2-1: The perceptron. The perceptron consists of n inputs $x_1, x_2, \dots, x_{n-1}, x_n$, weights w_1, \dots, w_{n-1}, w_n and the bias term w_0 . The inputs are multiplied with their respective weights and summed. The activation function σ uses this sum as input; its output is the final output of the perceptron, y . Adapted from: [2].

2-1-2 Multilayer perceptron

A Multilayer Perceptron Network (MLP) is created by connecting multiple perceptrons in a graph. The size of a MLP is defined by the number of hidden layers, the output layer L , and the size of each layer n_l where $l = 1, 2, \dots, L$. The input for a MLP is the same as for a single perceptron, defined as in Section 2-1-1: $\mathbf{x} = [x_1, x_2, \dots, x_{n_0}] \in \mathbb{R}^{n_0}$. The size of the output layer is defined as n_L and is chosen based on the desired output dimension of the network: $\mathbf{y} = [y_1, y_2, \dots, y_{n_L}] \in \mathbb{R}^{n_L}$. In Figure 2-2, we can see a MLP with $L = 3$ (2 hidden layers, 1 output layer), input size n_0 , layers sizes n_1, n_2 and output size n_3 .

Updating the mathematical description of a single perceptron in Equation 2-2 to a complete MLP can be done by defining the relations between the perceptrons as depicted in Figure 2-2. The output of a layer is taken as the input of the next. We define the network by applying Equation 2-2 to each hidden layer element (perceptron). Furthermore, we define

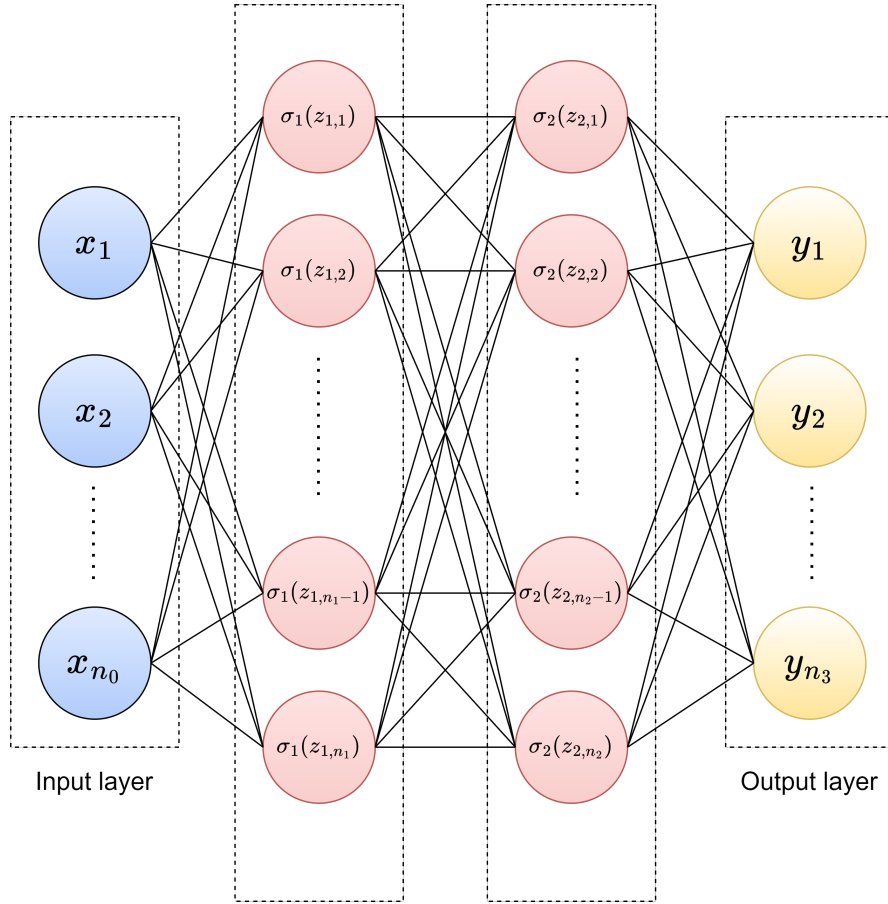


Figure 2-2: Depiction of a Multilayer Perceptron. The network has 1 input layer, 2 hidden layers and 1 output layer. The size of the input layer is n_0 , hidden layer 1 and 2 have sizes n_1 , and n_2 , respectively, and the output layer has size n_3 .

all parameters needed to build the network:

$$\mathbf{z}_i = [z_{i,1}, z_{i,2}, \dots, z_{i,n_i}], \quad i = 1, 2, \dots, L \quad (2-3)$$

$$\mathbf{b}_i = [w_{0,1}, w_{0,2}, \dots, w_{0,n_i}] \quad i = 1, 2, \dots, L \quad (2-4)$$

$$\mathbf{W}_i = \begin{bmatrix} w_{1,1} & \dots & w_{1,n_{i-1}} \\ \vdots & \ddots & \vdots \\ w_{n_i,1} & \dots & w_{n_i,n_{i-1}} \end{bmatrix} \in \mathbb{R}^{n_i \times n_{i-1}} \quad (2-5)$$

$$\theta_i = [\mathbf{W}_i \quad \mathbf{b}_i] \quad (2-6)$$

$$\mathbf{x} = [x_1, x_2, \dots, x_{n_0}, 1] \quad (2-7)$$

$$\theta = \{\theta_1, \theta_2, \dots, \theta_L\} \quad (2-8)$$

where \mathbf{z}_i is the vector of sums of layer L , also called the states of the layer, \mathbf{W}_i are the weights of layer i , θ is the combination of the weights and biases, and a one is added to the input vector to include said biases. The final output of the network is defined of the network

as a function of the parameters and the input:

$$\mathbf{y} = G(\theta; \mathbf{x}) = g_L(\theta_L; g_{L-1}(\theta_{L-1}; \dots g_2(\theta_2; g_1(\theta_1; \mathbf{x})))) \quad (2-9)$$

An example: two hidden layers, one output layer

We use Figure 2-2 as an example problem. When we apply Equations 2-1 to 2-9 on this problem, we get:

$$g_1(\theta_1; \mathbf{x}) = \sigma_1(\mathbf{z}_1) = \sigma_1(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1)$$

$$G(\theta; \mathbf{x}) = g_3(\theta_3; g_2(\theta_2; g_1(\theta_1; \mathbf{x})))$$

If we continuously apply the above, we get:

$$G(\theta; \mathbf{x}) = \sigma_3(\mathbf{W}_3 \sigma_2(\mathbf{W}_2 \sigma_1(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3)$$

We can generalize the above example for the output of the network as a function of all parameters θ , the network input \mathbf{x} and amount of layers L [42] and get Equation 2-9:

$$\mathbf{y} = G(\theta, \mathbf{x}) = g_L(\theta_L; g_{L-1}(\theta_{L-1}; \dots g_2(\theta_2; g_1(\theta_1; \mathbf{x})))) \quad (2-10)$$

where $\theta = \{\theta_1, \theta_2, \dots, \theta_L\}$ are the parameters in the network, $g_1(\theta_1; \mathbf{x}) = \sigma_1(\mathbf{z}_1)$ is the iteration for the first layer, L is the number of layers, including the output layer and \mathbf{y} is the output of the network.

2-1-3 Activation functions

An activation function defines the output of a perceptron based on the weighted and summed inputs, z . Activation functions introduce non-linearities into the network, allowing the model to learn more complex relationships in the data. For this reason, activation functions are also called the non-linearity of a NN.

The weights of a neural network are optimized using the gradient of its activation functions. Therefore, activation functions need to be differentiable. When calculating the gradient of an activation function, *the vanishing gradient problem* might arise. The vanishing gradient problem occurs when the gradient of the activation function becomes (close to) zero, slowing or stopping the weight optimization process. Activation functions with a derivative for large absolute values of z are more prone to this problem. An extensive explanation of the optimization process is given in Section 2-2-3.

Commonly used activation functions are the sigmoid-, hyperbolic tangent- and Rectified Linear Unit (ReLU) activation functions [41, 43]. The sigmoid activation function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (2-11)$$

The sigmoid function takes the summed output of the perceptron z , and maps it to a value between zero and one, yielding the bounded output: $\sigma(z) \in (0, 1)$. Large negative values of z are pushed close to zero, whereas large positive values of z are pushed close to one. Due to

this range, the sigmoid function is often used for networks where a probability prediction is the output. A con of the sigmoid function is that it is prone to the vanishing gradient problem.

The hyperbolic tangent function is defined as:

$$\sigma(z) = \frac{\sinh(z)}{\cosh(z)} = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (2-12)$$

This function is an alternative to the sigmoid function, where the range is given by $\tanh(z) \in (-1, 1)$. It is mostly used in two-class classification problems, where the output is assigned to one of two classes. The hyperbolic tangent function is centered around zero, making it more stable than the sigmoid function but also more sensitive to the initialization of the weights. The function also experiences the vanishing gradient problem.

The ReLU activation function is described as:

$$\sigma(z) = \max(0, z) = \begin{cases} z & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases} \quad (2-13)$$

The ReLU function maps negative values to zero, resulting in a piecewise linear function $\sigma(z) \in [0, \infty)$. The function is computationally efficient because it only requires a threshold operation. It avoids the vanishing gradient problem as with the sigmoid or hyperbolic tangent function since the derivative will not get close to zero for large values of z . The ReLU, however, can suffer from the dying ReLU problem. In this case, the z value of too many neurons becomes smaller than zero, giving a zero output after computation using the ReLU. Neurons can get stuck in this situation and are no longer updated when training the neural network [44].

In Appendix A, a summary of these activation functions, their derivatives, and visualizations are shown.

2-1-4 Function approximation using Neural Networks

MLPs are used to approximate functions using historical or observable data. In 1989, Cybenko proved with the universal approximation theory that a NN can approximate a function [45]:

Let $X \subset R^m$ be a compact set and $g : X \rightarrow R$ a continuous function. Assume sigmoid activation functions. Then, for any error threshold $\epsilon > 0$, there exists a number $K > 0$ such that there exists a neural network f with number of hidden neurons $k > K$. This relation can be expressed as

$$\forall x \in X, |f(x) - g(x)| \leq \epsilon$$

This theory essentially states that as long as we have no constraints on the number of perceptrons (nodes) in our single-layer network and the weights can be of unlimited size, a single-layer network with continuous sigmoidal functions can approximate continuous functions with arbitrary precision.

2-2 Parameter estimation

To estimate the set of parameters $\theta = \{\theta_1, \theta_2, \dots, \theta_L\}$ of a NN, called *training the network*, a loss function needs to be quantified and an optimization method chosen. The optimization method used to train a NN is a backpropagation algorithm with gradient descent.

2-2-1 Loss function

The loss function is the error that is minimized to train and optimize a NN. A common method is to measure the difference between the output of the network, $\hat{\mathbf{y}}$, and the expected output, \mathbf{y} , using the Mean Squared Error (MSE) [46]:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2 \quad (2-14)$$

For this loss function, the goal would be to minimize the squared difference between our estimated and measured outputs. The function to be minimized is called the objective function. For a MLP, the function would be [41]:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathcal{L}(\theta) = \underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^N (y_i - G(\theta; \mathbf{x}_i))^2 \quad (2-15)$$

where θ^* are the optimal parameters of the network, θ are the parameters to be optimized, \mathbf{x} are the inputs to the network, y_i is the ground truth output and $G(\theta; \mathbf{x}_i)$ is the output generated by the network.

Within NN, a great variety of loss functions or combinations thereof can be chosen. In the rest of this thesis, loss functions of specific NNs will be explained when the network itself is discussed.

2-2-2 Gradient descent

Gradient descent is a first-order iterative optimization algorithm that uses weight updates to comprise a small step in the direction of the negative gradient [47]. For each iteration, the weights will be updated toward the negative gradient:

$$\mathbf{w}_{(\tau+1)} = \mathbf{w}_{(\tau)} - \eta_{\tau} \nabla \mathcal{L}(\mathbf{w}_{\tau}) \quad (2-16)$$

where $\mathbf{w}_{(\tau+1)}$ is the next iteration, $\mathbf{w}_{(\tau)}$ is the current iteration, $\mathcal{L}(\mathbf{w}_{\tau})$ is the loss function and η_{τ} is the learning rate at the current iteration. After an update, the gradient is re-evaluated for the new weight and the process is started all over. Since this function is updated over all the data, the entire set of weights, inputs and outputs would need to be processed to evaluate each gradient $\nabla \mathcal{L}$. This is called *Batch Gradient Descent*.

Due to the utilization of all data in each iteration step of batch gradient descent, this method is sub-optimal for training neural networks because it slows down with increasing data size. For

this reason, *Stochastic Gradient Descent* is introduced [42, 47]. Stochastic gradient descent makes an update to the weights based on the data points of the specific iteration, updating Equation 2-16 to:

$$\mathbf{w}_{(\tau+1)} = \mathbf{w}_{(\tau)} - \eta_{\tau} \nabla \mathcal{L}_n(\mathbf{w}_{\tau}) \quad (2-17)$$

where all parameters are the same as in Equation 2-16, except the addition of the single data point based loss function \mathcal{L}_n [48]. The stochastic gradient descent algorithm can now online train the NN, handling redundancy in the data more efficiently. Another advantage is that the stochastic gradient descent method escapes a local minimum more easily by only using part of the data [47].

The final parameter we can tune is the learning rate. This parameter is important to influence the training time of the network and its ability to converge to the optimum. The learning rate determines how large the step taken in the direction of the gradient is. It is a trade-off between the rate of convergence and overshooting. A high learning rate can make the optimization pass the minimum, while a low learning rate can get the optimization stuck in local minima or elongate the training time. One of the stochastic gradient descent algorithms with decaying learning rate, ADAM, is shown in pseudo code in Appendix A.

The stochastic gradient descent can be further updated by adding momentum. This updates Equation 2-17 to [49]:

$$\mathbf{v}_{(\tau+1)} = \gamma \mathbf{v}_{(\tau)} - \eta_{\tau} \nabla \mathcal{L}_n(\mathbf{w}_{\tau}) \quad (2-18)$$

$$\mathbf{w}_{\tau+1} = \mathbf{w}_{\tau} + \mathbf{v}_{\tau+1} \quad (2-19)$$

where γ is the momentum and is generally set $\gamma \approx 0.9$ [49]. The momentum term increases the speed of convergence of the stochastic gradient descent algorithm by including the previous weight update. The momentum term increases if the gradients point in the same direction. It decreases the update for gradients that point in opposing directions [48].

2-2-3 Backpropagation

Backpropagation is the algorithm used by a NN to train and learn its weights [41]. Since hidden layers have no target output, it is hard to create an error function specific to each node. Error functions of separate nodes depend on the values of the previous layer. The coupling of parameters between layers can make gradient calculations slow and the math complicated. By backpropagating, the speed of the gradient calculation can be increased. We start at the output layer of our network and calculate the error toward the input.

Backpropagation requires three things: A dataset of input-output pairs $(\mathbf{x}_i, \mathbf{y}_i)$, a network with parameters θ , containing the weights and biases of the network, as discussed in Section 2-1-2, and a loss function $\mathcal{L}(\theta, \mathbf{x})$ as described in Section 2-2-1. Rewriting equation 2-17 with these parameters gives the stochastic gradient descent algorithm:

$$\theta_{\tau+1} = \theta_{\tau} - \eta_{\tau} \frac{\partial \mathcal{L}(\theta_{\tau}, \mathbf{x})}{\partial \theta} \quad (2-20)$$

where τ is the iteration step. The backpropagation algorithm is used to determine the gradient of the loss function for all weights. Using the MSE from Equation 2-14 as example, the partial

derivative can be further expanded as:

$$\frac{\partial \mathcal{L}(\theta, \mathbf{x})}{\partial \theta_i} = \frac{1}{N} \sum_{j=1}^N \frac{\partial}{\partial \theta_i} \left((\hat{y}_j - y_j)^2 \right) \quad (2-21)$$

where i is the layer and j is the data point. For the error term from the output to the last hidden layer, this is straightforward [2]:

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta_L} = \frac{\partial \mathcal{L}(\theta)}{\partial \mathbf{y}_j} \frac{\partial \mathbf{y}_j}{\partial \theta_L} \quad (2-22)$$

To go back two layers, the chain rule is applied:

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta_{L-1}} = \frac{\partial \mathcal{L}(\theta)}{\partial \mathbf{y}_j} \frac{\partial \mathbf{y}_j}{\partial \sigma_{L-1}(\mathbf{z}_{L-1})} \frac{\partial \sigma_{L-1}(\mathbf{z}_{L-1})}{\partial \theta_{L-1}} \quad (2-23)$$

We can continue this for all layers to update the weights of all layers. The steps of training a NN can be summarized as:

- (a) Make a forward pass through the network to calculate the loss.
- (b) Using the backpropagation algorithm in combination with the chain rule to calculate the losses of the individual layers and their weights.
- (c) Use the gradient descent method to update the weights.
- (d) Repeat until the stopping criterion is reached.

2-3 Specialized Neural Networks

2-3-1 Convolutional Neural Network

In imaging, a common choice of NN is the CNN [32]. CNNs are considered to be fundamental in deep learning methods, especially in image analysis and segmentation procedures [50]. CNN utilise the same backpropagation algorithm as MLP to update their weights. CNNs can get increasingly complex. Still, they have the same essential building block: a convolutional encoder that uses convolutional and pooling layers to encode the image into a feature representation. This feature representation can be used to predict what is in an image or be decoded to make for pixel-wise labeling (assigning each pixel to a specific class). An example of a CNN is shown in Figure 2-3.

Convolutional operation

A convolutional operation is performed by passing a kernel $K \in \mathbb{R}^{k_1 \times k_2}$, also called the window over the input image $I \in \mathbb{R}^{M \times N}$, where $M \times N$ is the image size. The kernel is moved over the image with stepsize s , also called the stride. An example of this process is shown in Figure 2-4. The size of the output convoluted image is $S \in \mathbb{R}^{\left(\frac{k_1 - J}{s} + 1\right) \times \left(\frac{N - k_2}{s} + 1\right)}$.

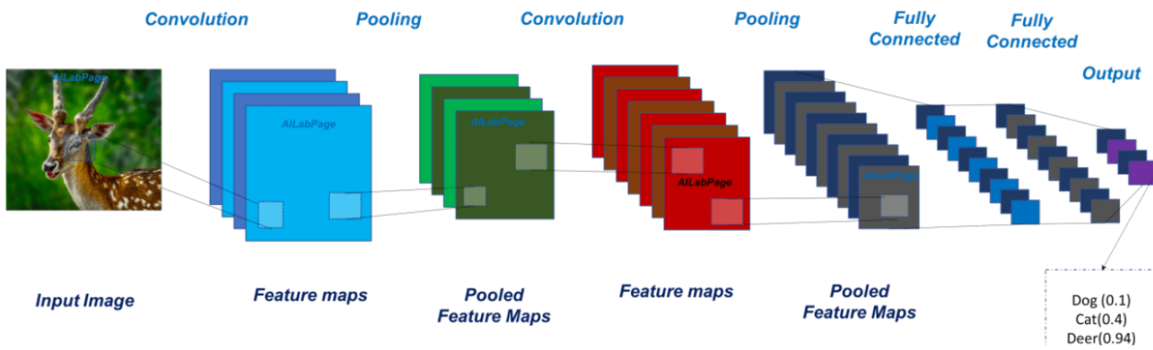


Figure 2-3: Schematic overview of a Convolutional Neural Network. An input image is convolved into several feature maps. The feature maps are then pooled for downsizing and feature selection. This process is repeated until the desired feature space is calculated. After the determination of the feature mapping, the feature map is passed through several fully connected layers to predict what animal is presented to the network. The output is a probability for each possible animal class (Dog, Cat and Deer). Adapted from: [3]

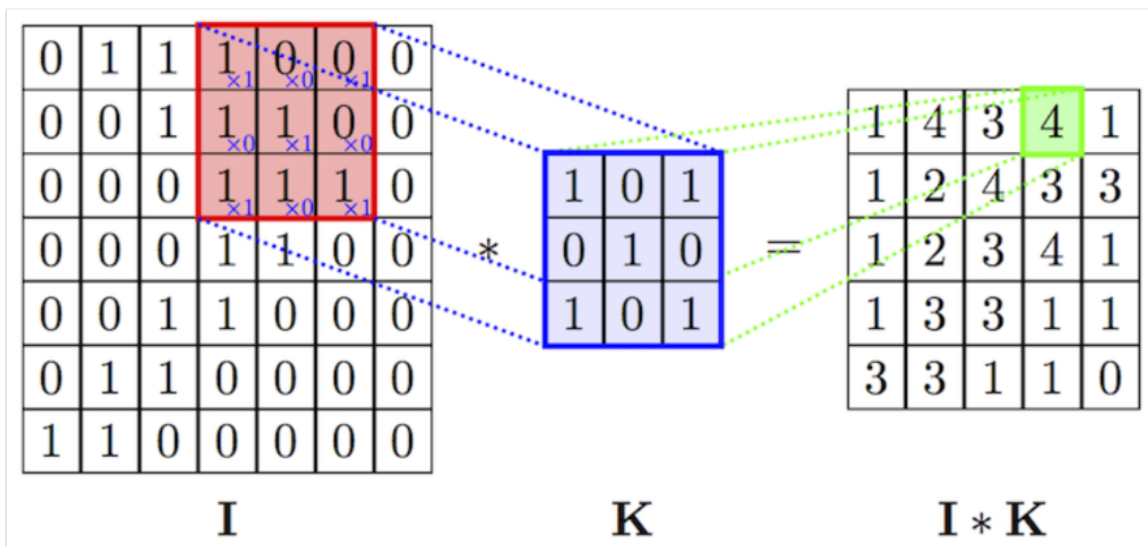


Figure 2-4: Convolutional operation on I with kernel K with stride $s = 1$. The kernel (blue) is moved over the image I, starting in the left top corner. By moving the kernel one panel to the right (stride=1) while calculating at each intermediate step, we get the convolutional mapping $I*K$. From: [4]

Mathematically, we can describe the convoluted image S as:

$$S[m, n] = (K * I)[s(m - 1) + 1, s(n - 1) + 1] = \sum_j \sum_k I[s(m - 1) + j, s(n - 1) + k]K[j, k] \quad (2-24)$$

where $j = 1, 2, \dots, k_1$, $k = 1, 2, \dots, k_2$ and s is the stride.

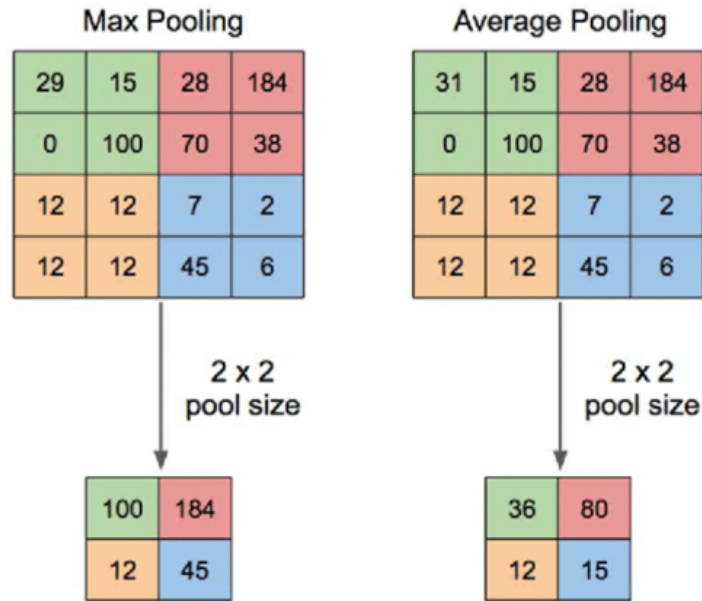


Figure 2-5: Pooling of a 4x4 matrix with pool size 2x2. Adapted from: [5]

Pooling operation

The pooling operation is used to decrease spatial size by selecting the most prominent features or taking the average of the present features. The most common pooling approaches are max pooling and average pooling. An example of both is shown in Figure 2-5.

Max Pooling takes the largest value from a pool of size $p_1 \times p_2$. The largest value is stored and used for further iterations. The maximum value is calculated according to:

$$P[n, m] = \max S[p_1(n-1) + i, p_2(m-1) + j]$$

where S is a 2D image (or one of its convolutions) of size $h \times w$, $i = 1, 2, \dots, p_1$, $j = 1, 2, \dots, p_2$, $n = 1, 2, \dots, N$ and $m = 1, 2, \dots, M$. Furthermore, the size of the pooling matrix is determined by the size of the original image and the size of the pooling window: $p_1 N = h$ and $p_2 M = w$.

Average Pooling takes the average from all values in a pool and rounds it down to the nearest integer according to:

$$P[n, m] = \sum_{i=0}^{p_1-1} \sum_{j=0}^{p_2-1} [S[p_1(n-1) + i, p_2(m-1) + j]]$$

where the parameters are the same as for max pooling.

Fully connected layers

After the convolutional and pooling layers, we are left with a feature representation of our image. This feature representation, called the latent space, is then processed by one or more fully connected layers, as shown in Figure 2-3. These fully connected layers have a similar structure to the MLP explained in Section 2-1-2.

2-3-2 Encoder-decoder network structure

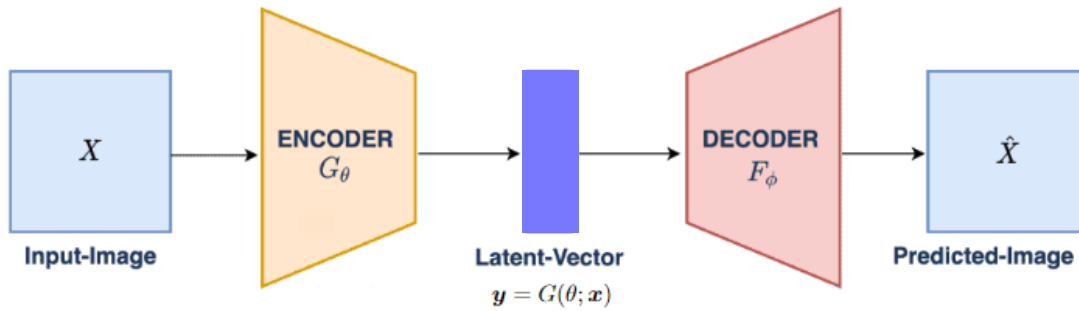


Figure 2-6: Example of an encoder-decoder network structure. The encoder utilises a set of pooling and convolutional layers to generate the lower dimensional latent space of the input image. The decoder takes the latent space as input and generates a reconstruction of the input image X . Adapted from: [6]

The encoder-decoder structure is a commonly used structure in many Deep Learning (DL) solutions, such as language processing networks [51], fingerprint recognition [52], object detection [53], and image classification [13]. The encoder-decoder structure is based on two stages. First, an encoding stage, where the information of the input is downsized into a lower dimension, called the latent space. Second, a decoder transforms this latent space into a reconstruction of the original input. In Figure 2-6, the structure of an encoder-decoder network for image processing is shown. The encoder takes in an image and processes it through a series of convolutional and pooling layers to extract features and reduce the dimensionality of the data using a convolutional approach, as explained in the Section 2-3-1. Instead of processing the latent space with several hidden layers to generate an output, the latent space is used as input for the decoder. A decoder uses an unpooling method to increase the dimension of the latent space and convolutional or transposed convolutional layers to process the output of the unpooling layers [54].

Unpooling

Unpooling is an upsampling method used to increase the dimensions of the feature space. A commonly used method for upsampling is switch variable unpooling [9]. The indices of max pooling indices explained in Section 2-3-1, are stored as switch variables. These switch variables are used to replace the value in the location that it originally had before the pooling operation. The size of the feature space after a switch variable unpooling layer is equal to the size of its paired pooling layer, which originally stored the switch variables. The unpooling process is further illustrated in Figure 2-7.

An alternative to switch variable pooling is Bed-of-Nails unpooling. In this situation, the original locations of the values are not stored, but a 2D feature space is expanded by adding p_b zeros between the elements of a feature space to increase the dimensions. Where switch variable unpooling has an output of the same size as its matched pooling layer $\mathbb{R}^{n \times m}$, Bed-of-Nails unpooling dimensions can be chosen based on the amount of added zeros p_b between the elements: $\mathbb{R}^{(p_b+1)n \times (p_b+1)m}$. The Bed-of-Nails pooling method is depicted in Figure 2-8.

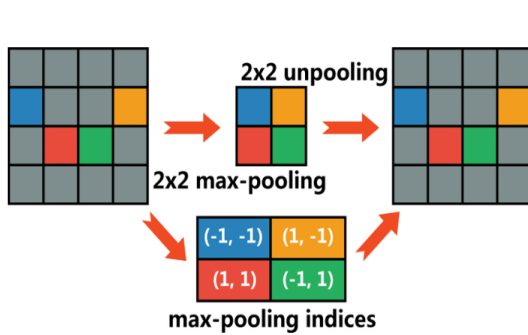
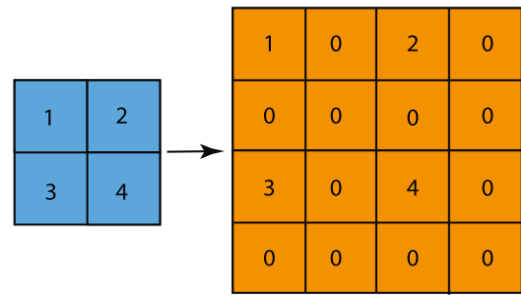


Figure 2-7: Switch variable unpooling of a pooled image with a 2×2 window. The stored indices are used to place an element at its original index before the pooling step. From: [7]



Input: 2×2 Output: 4×4

Figure 2-8: Bed-of-Nails unpooling. Zero values are placed between the elements of a 2D feature space with spacing $p_b = 1$. From: [8]

The output of both unpooling methods is sparse. To increase the information density of the 2D matrices, either a convolutional layer, as explained in Section 2-3-1, or a transposed convolutional layer is used [54].

Transposed convolutional layers

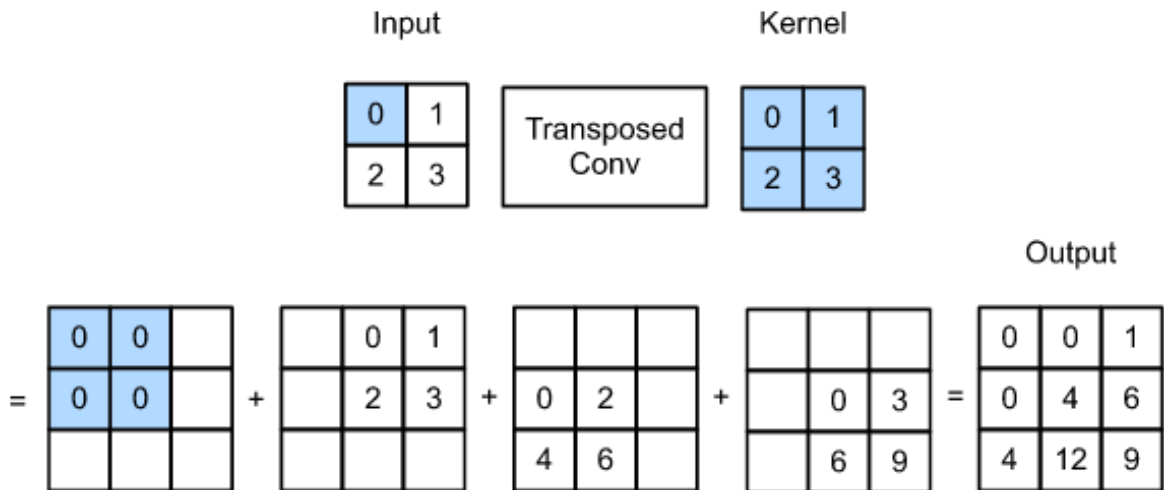


Figure 2-9: Example of the transposed convolution step with stride $s = 1$ and padding $p = 0$. Each element of the input is multiplied with each element of the kernel and stored. In blue, the example of the multiplication of the first input element with the kernel is shown. The outputs of all the multiplications are summed. From: [9]

Transposed convolution layers are used to increase the density of information of the feature space obtained by unpooling through transposed convolution operations with learned kernels, and can be used for applications such as creating a dense map of a sparse matrix [9] or super-resolution [55]. Where convolution combines the information of multiple matrix elements into

one element, transposed convolution is the opposite process. It takes the information of a single element and spreads it over multiple elements using kernel $K \in \mathbb{R}^{k \times k}$. The output of the transposed convolution layer is a dense map of the originally sparse output of the unpooling layer [9]. The transposed convolution process is illustrated in Figure 2-9

In some applications, the transposed convolutional layer is applied to increase the dimensions of a 2D feature space directly without using an unpooling layer by setting the stride $s > 1$ [34]. The generated output dimensions of the transposed convolutional layer are described as:

$$O_o = s(O_i - 1) + O_k - 2p \quad (2-25)$$

where O_o is the output size, O_i is the input size, O_f is the filter size, O_k is the kernel size and p is the padding.

2-3-3 Generative Adversarial Networks

GANs are mostly used for unsupervised learning [56]. The aim of a GAN is to learn the hidden or underlying structure of the data [11]. The training data is used to learn a distribution, and the model tries to learn the distribution.

A GAN consists of two parts, the generator and discriminator, represented in Figure 2-10. The generator tries to find a mapping $G : z \rightarrow x$ from a D dimensional noise vector. The

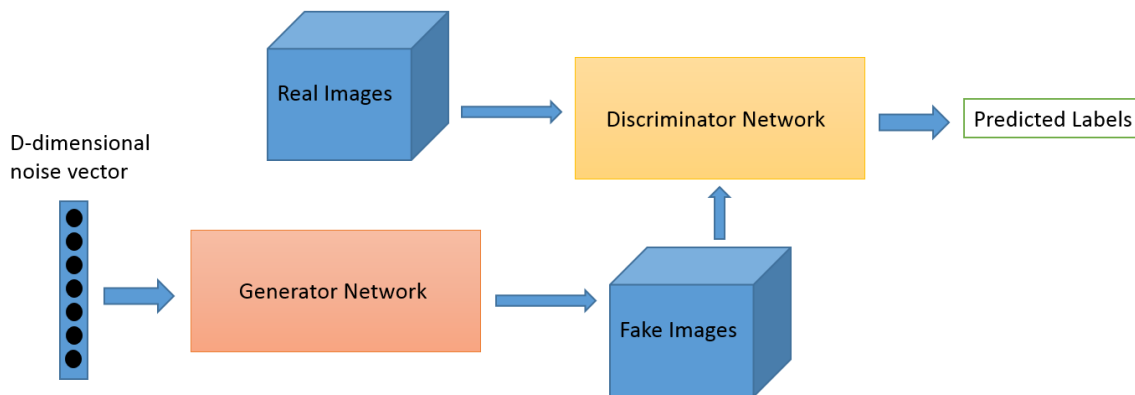


Figure 2-10: Structure of a Generative Adversarial Network. A noise vector is fed to the generator, that creates a fake image from this noise vector. The discriminator predicts if a fed image is "fake" or "real". From: [10]

discriminator tries to label samples from the original dataset as real, and images generated by the generator as fake. Essentially, the discriminator and generator are playing a game where they try to fool each other: the generator tries to make images that the discriminator labels as real. As stated in [11], the discriminator and generator are playing a two-player min-max game, where the generator tries to minimize $\log(1 - D(G(z)))$, while the discriminator tries to maximize the number of correct labels $D(G(z))$ and $D(x)$. we can mathematically formulate this as:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \quad (2-26)$$

where G is our generator, D is the discriminator, x is our data and z is our noise vector. The steps taken towards optimization of a GAN are shown in Figure 2-11. The simultaneous

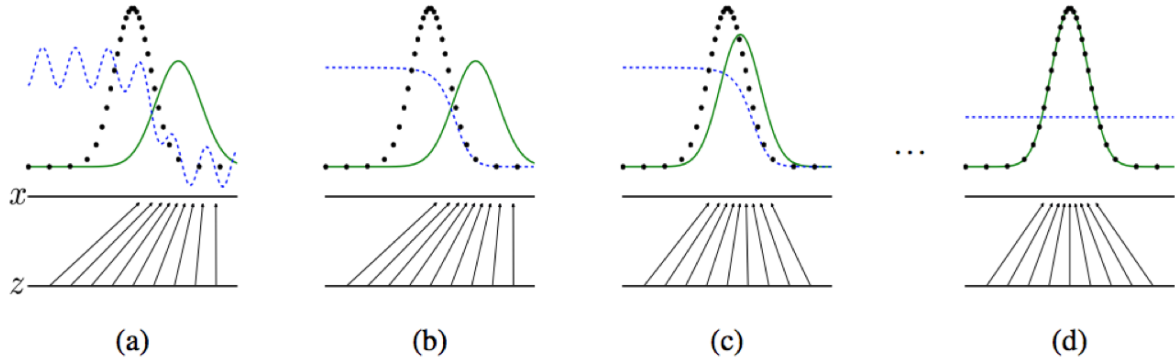


Figure 2-11: Generative Adversarial Network trained simultaneously: we can see the discriminative distribution (blue dotted line) the data generating distribution (black dotted line) p_x and the generative distribution (green line) $p_g(G)$. The lower horizontal line is the domain from which z is sampled. The upper horizontal line is part of the domain of x . The arrows between the lower and upper horizontal lines show how the mapping $x = G(z)$ imposes the non uniform distribution p_g on the transformed samples. From: [11]

update in Figure 2-11 can be explained in 4 steps.

In (a), we can see an adversarial pair, a discriminator, and a generator close to convergence. This means that the distributions p_g and p_{data} are almost the same. The discriminator is a partially accurate classifier in this situation.

In (b), the inner loop of the algorithm is trained to discriminate samples from data:

$$D^*(\mathbf{x}) = \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + p_g(\mathbf{x})}$$

In (c), after an update to the generator, the gradient of the discriminator is used to "push" $G(z)$ to regions that are more likely to be classified as data.

In (d), the discriminator is no longer able to tell the difference between both data distributions and the blue dotted line has reached $D(\mathbf{x}) = \frac{1}{2}$. This means that the distributions of the data and generator are the same: $p_g = p_{data}$. This happens if there is enough capacity and after enough training.

2-4 Bias, variance, overfitting and underfitting in Neural Networks

Overfitting and underfitting are terms in ML describing a too tightly or loosely fitted model on the training data by the ML algorithm. **Overfitting** occurs when a statistical model is fitted too tightly on the training data. This is good for the accuracy on the training data, but on unseen data the model cannot perform accurately. There are different reasons for overfitting, like an complicated model overtrained on noise or a (too) complex model trained on small data sets [47]. **Underfitting** occurs when a statistical model is fitted too loosely on the training data and the model is not able to capture the relation between input and output. We have created a too simple model to capture the complexity of the data. Since a generalised model allows ML algorithms to make predictions and classify data, neither overfitting or underfitting are desired.

The bias and variance are other important basic principles in ML. **Bias** is the term used for the difference between the prediction of the values by the ML model and the correct value. A high bias gives a large error in training and testing. If the bias is high, a model might be too simple for the complexity of the data. **Variance** is the variability of the model prediction on the data. A model with a high variance has a complex fit to the training data and might not be able to fit unseen data correctly. As a result, models with a high variance can perform good on the training data, but have a high error on the test data. Mathematically, we can define the bias and variance as [47]:

$$\text{Bias}(\hat{y}) = E[\hat{y}] - y, \quad \text{Var}(\hat{y}) = E[(E[\hat{y}] - \hat{y})^2] \quad (2-27)$$

Where y is the solution to $f(x)$ and \hat{y} is the solution to the estimated function $\hat{f}(x)$.

Bias, variance, underfitting and overfitting are all related. Models with low variance have a high bias and are underfitted on the data. Models with low bias and high variance are overfitted on the data. To find an optimal model, we need to find the balance between the bias and variance, since we cannot minimize both. This problem is known as the **Bias-Variance Trade-Off**. In 2-12, the relation between bias, variance, underfitting and overfitting is shown.

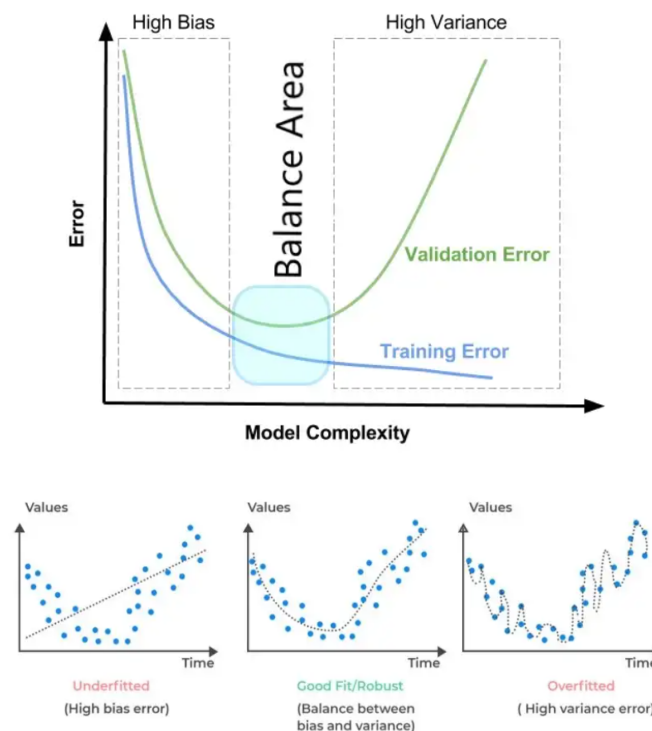


Figure 2-12: The relation between bias, variance, underfitting and overfitting. A high bias means underfitting on the training data, while high variance shows overfitting on the training data. A model that shows a good fit to the training data, has a balanced bias and variance. This is called the Bias-Variance Trade-Off. Adapted from: [12]

Chapter 3

Related works

During this thesis, several Neural Networks (NNs) have been used. In the previous chapter, we have seen the theory behind NNs and some specific network structures. In this chapter, the networks that have been used in this thesis are discussed. To understand what these networks do and how they behave, their network structures and parameters will be discussed. Furthermore, the datasets used by the networks are discussed.

3-1 Semantic segmentation

In image processing, image segmentation is the process of dividing an image into multiple segments linked to a particular class. A sub-process of image segmentation is semantic segmentation, where each pixel is labeled and assigned to a class. In this section, we will discuss the SegNet NN for semantic segmentation, and its parameters [13]. The general architecture is discussed first. Then, we discuss the computations and network specifics not addressed in Chapter 2. The mathematics of the loss function is discussed, and the CamVid semantic segmentation dataset is shown.

3-1-1 Segmentation architecture

The SegNet encoder-decoder architecture for semantic segmentation has similar properties as encoder-decoder networks described in Section 2-3-2. A softmax pixel-wise classification layer follows the encoder-decoder structure. The encoder is based on the VGG16 encoder network [57]. The fully connected layers for classification are dropped, leaving 13 convolution and pooling layers that create the latent space. Each encoder layer has a matching decoder layer, generating an image of the same size as the input. The output of the final convolutional layer of the decoder is passed to a multi-class softmax layer that outputs a pixel-wise segmented image.

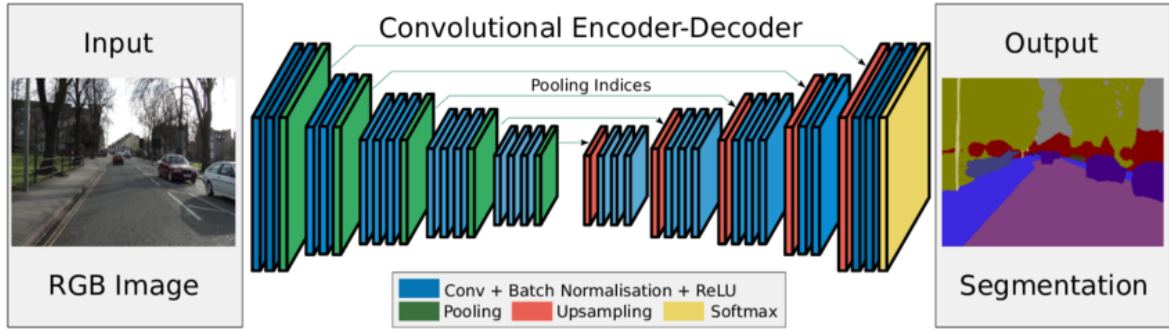


Figure 3-1: The SegNet NN architecture. The SegNet architecture is based on an encoder-decoder structure. An input image is encoded using convolutional and max pooling layers with switch variables, storing the pooling indices. The resulting latent space is upsampled using switch variable unpooling and convolutional layers. The last layer is a soft-max classifier layer of the same size as the RGB input image, assigning each pixel to a class and generating a pixel-wise segmented image. From [13].

Each encoder convolutional layer performs convolution with filters that generate an accompanying feature map. Each feature map is batch normalized, and a Rectified Linear Unit (ReLU) activation function is applied to add non-linearity, as explained in Section 2-1-3. The pooling layers that follow a convolution layer are max pooling layers that store the switch variables as in Section 2-3-2. The pooling kernels have size 2×2 with stride $s = 2$.

The latent space generated by the encoder is upsampled using a combination of unpooling layers using the switch variables as in Section 2-3-2 and convolutional layers. The output of the decoder is processed with a softmax layer that predicts a class (out of 12) for each input pixel, generating a pixel-wise segmented image.

Batch normalization

Batch normalization is a method used to train NN faster and more accurately by normalizing the output of a layer before processing it with the next layer [58]. The activation vectors of hidden layers are normalized using the mean and variance of the current batch. The mean μ_i and variance σ_i^2 of layer i are calculated using [59]:

$$\mu_i = \frac{1}{N_i} \sum_{j=1}^{N_i} z_{i,j} \quad (3-1)$$

$$\sigma_i^2 = \frac{1}{N_i} \sum_{j=1}^{N_i} (z_{i,j} - \mu_i)^2 \quad (3-2)$$

where z_i are the states of layer i according to Equation 2-3 and N_i are the numbers of neurons in layer i . The mean and variance are used to normalize each state of layer i according to:

$$z_{\text{norm}(i,j)} = \frac{z_{i,j} - \mu_i}{\sqrt{\sigma_i^2 - \epsilon}} \quad (3-3)$$

where ϵ is a constant used for numerical stability. The normalised state vector of layer i is defined by all state elements: $\mathbf{z}_{\text{norm}(i)} = [\mathbf{z}_{\text{norm}(i,1)}, \mathbf{z}_{\text{norm}(i,2)}, \dots, \mathbf{z}_{\text{norm}(i,N_i)}]$. The normalized states are used to compute the next layer as explained in Section 2-1-2.

Softmax classification layer

The softmax classification layer is a trainable classifier that outputs the probability of each class per pixel. The pixel is then assigned to the class with the highest probability. Softmax classification is used as an activation function in the final layer of SegNet and predicts the probability that an output belongs to class k . The softmax function is defined for input \mathbf{x} and output \mathbf{y} as [60]:

$$p(\mathbf{y}_i = k \mid \theta, \mathbf{x}_i) = g(\mathbf{z}_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (3-4)$$

where θ are the weights, \mathbf{z} is the state vector that is the size of the classes, and $k \in K$ is the number of classes.

Loss and activation functions

The cross-entropy loss function is used as the loss function for SegNet. The cross-entropy loss function is a common choice as a loss function in neural networks for image segmentation [61]. The multi-class cross-entropy loss is defined as:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{i=1}^N \sum_{k=1}^K y_{i,k} \log \hat{y}_{i,k} \quad (3-5)$$

where $\hat{y}_{i,k}$ is the probability for pixel i belonging to class k and $y_{i,k}$ is either 0 or 1 and called the truth value, stating if k is the correct class. The loss function is minimized according to the gradient descent method with backpropagation as in Section 2-2-2 and Section 2-2-3.

CamVid dataset

The CamVid dataset is an urban-based dataset obtained with a 3CCD Panasonic HVX200 digital camera [62]. The dataset consists of 367 training, 101 validation, and 233 test images. The image size is 480x360 and contains 12 classes. A subset of images of the CamVid dataset is shown in Figure 3-2. One of the labels of the CamVid dataset is "Unlabelled". This label is used when an object does not fit in one of the eleven other classes.

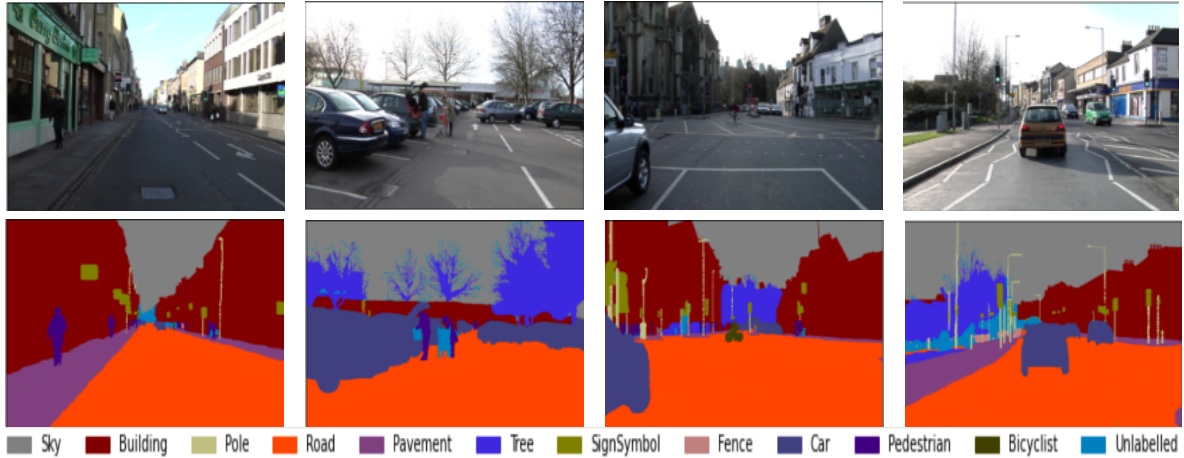


Figure 3-2: An example of the images in the CamVid dataset. The top row shows the scenery images taken with the camera. The bottom row shows the Ground Truth segmented images paired with the top row. The legend shows the twelve classes of the CamVid dataset.

3-2 Dual Hierarchical Aggregation Network

The Dual Hierarchical Aggregation Network (DHAN) is a network created for feature removal. The network takes an image with shadow and outputs a shadow-free image and respective mask. Figure 3-3 shows a schematic overview of the DHAN. In this section, the general architecture of the DHAN is discussed. The separate elements of the DHAN architecture, one through four in Figure 3-3, are discussed in depth after the architecture overview. Finally, the datasets for shadow removal are discussed.

3-2-1 Architecture

The DHAN takes shadow image I_s as input and outputs a shadow-free image I_f and the respective mask M . The backbone of the network is based on the VGG16 network [63]. The VGG16 network is discussed in Section 3-1. The DHAN is the generator in a Generative Adversarial Network (GAN). The discriminator is a plane discriminator as in Section 2-3-3. The DHAN generator network structure is shown in Figure 3-3.

In **(1)**, the image I_s is processed by the VGG16 network, and the feature space after each pooling layer is stored. In **(2)**, these feature spaces are processed with two convolutional layers with a 1×1 kernel. Each convolutional layer follows a LeakyReLU layer. After the convolutional layers, several dilated convolution layers with dilation rate k follow. The dilation rate increases with a factor of two for each next dilated convolution layer. The first dilated convolution layer takes the output of the second convolution layer as input. The dilated convolution layers with $k > 1$ take the output of the aggregation nodes (dilation rate $k = 2, 8, 32$) or the previous dilation layer (dilation rate $k = 4, 16, 64$) as input, as shown in Figure 3-3. In **(3a)** and **(3b)**, attentive aggregation is applied in four nodes, taking the output of the previous two convolution layers and attentive aggregation node as input (if there is a prior attention aggregation node). The inputs to each node are specified in Figure 3-3 with arrows.

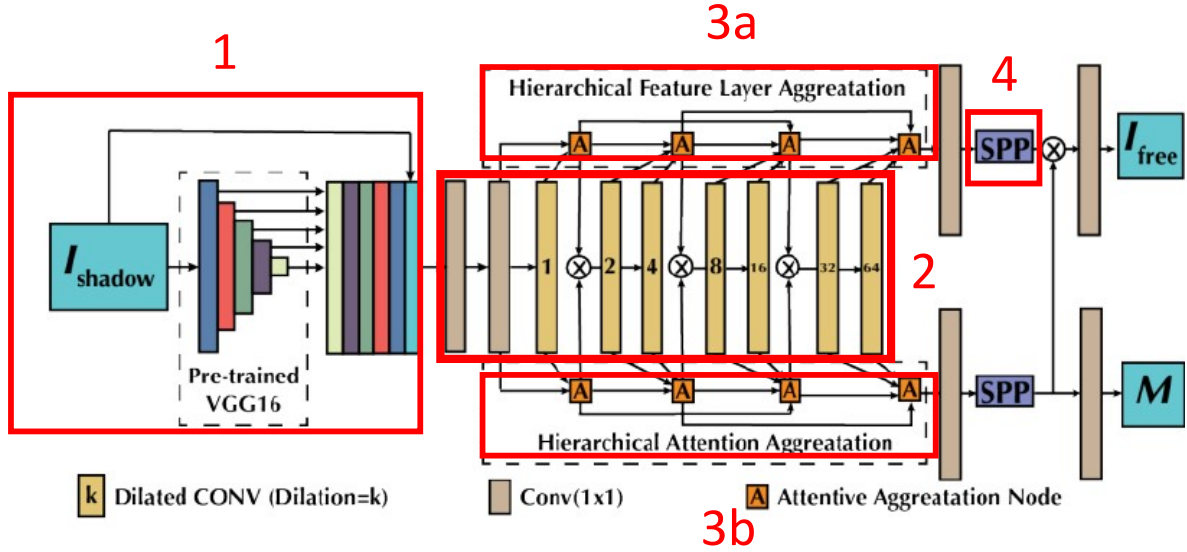


Figure 3-3: Structure of the Dual Hierarchical Aggregation Network. A shadow image is encoded using the VGG16 network (1). The encoded features are passed through two convolutional layers of convolution size 1x1 and processed further by several dilated convolution layers and attentive aggregation nodes (2). The final two attentive aggregation nodes' results are processed by a convolutional layer, spatial pooling pyramid (4), and convolutional layer. The final outputs are a shadow-free image and its respective shadow mask. Adapted from: [14].

A convolutional layer processes the output of the final two aggregation nodes with a 1x1 kernel. In (4), a spatial pooling pyramid is applied, followed by another convolutional layer, generating a shadow-free image I_f and the shadow mask M .

Activation function LeakyReLU

The LeakyReLU activation function is a variant of the ReLU activation function discussed in Section 2-1-3. The LeakyReLU incorporates information from input values smaller than zero by incorporating a small slope α to negative input values. The LeakyReLU function is mathematically described as follows:

$$g(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases} \quad (3-6)$$

The rule of thumb is $\alpha < 0.02$ [64].

3-2-2 Dilated convolution

Dilated convolution captures information stored in elements of a 2D array that are spatially further apart by increasing the receptive field. This means that information from a wider range is processed. The process is the same as convolution, except that kernel values K are moved over the input spaced by dilation rate d . We update Equation 2-24 with this dilation factor:

$$S[m, n] = (K * I)[s(m-1)+1, s(n-1)+1] = \sum_{l_1} \sum_{l_2} I[s(m-1)+l_1, s(n-1)+l_2] K[j, k] \quad (3-7)$$

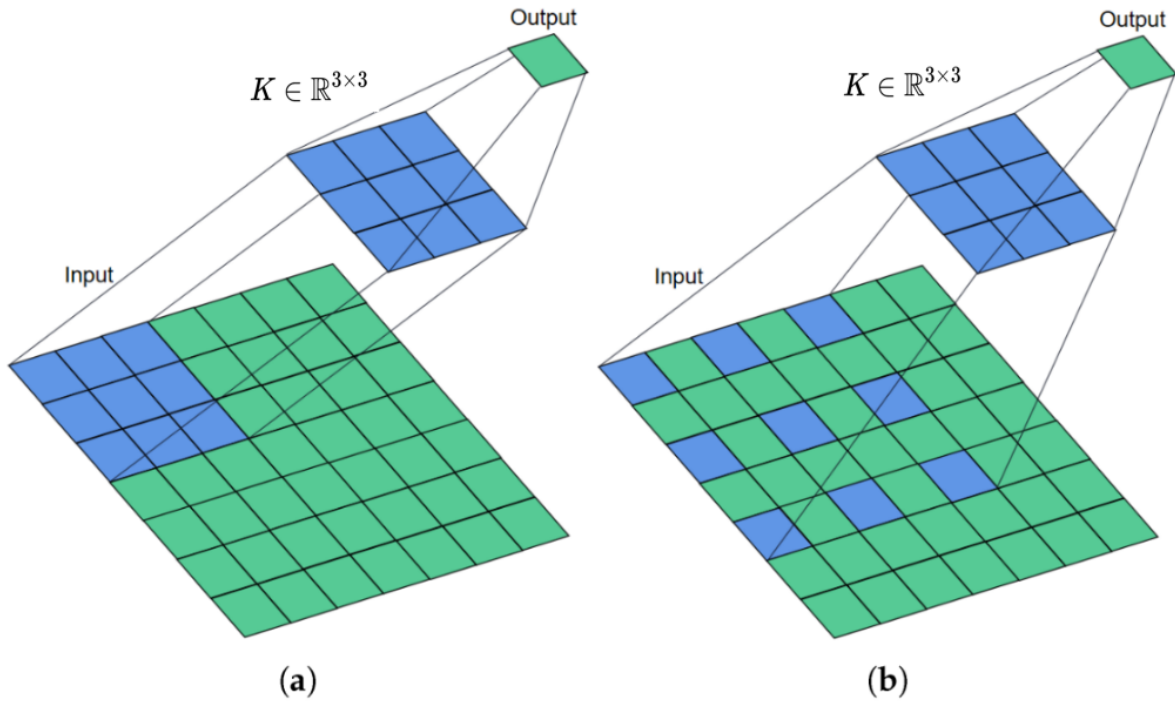


Figure 3-4: Schematic overview of dilated convolution. **(a)** Shows dilated convolution for $d = 1$, which is the same as normal convolution. **(b)** Shows dilated convolution with dilation rate $d = 2$. The kernel is moved over the input with spacing $d - 1$, increasing the spacing between the matrix convoluted with kernel K . Adapted from: [15]

where $l_1 = d(j - 1) + 1$ and $l_2 = d(k - 1) + 1$, with d the dilation rate. We can deduce that dilated convolution with a dilation rate $d = 1$ is the same as convolution since Equation 3-4 is equal to Equation 2-24 for $d = 1$.

3-2-3 Attentive aggregation nodes

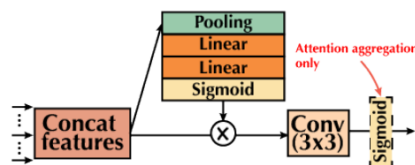


Figure 3-5: Attentive Aggregation Node in the DHAN. The features of several layers or nodes are concatenated. This concatenated feature space is then pooled and passed through three fully connected linear layers, layers one and two containing linear (identity) activation functions and the final layer having a sigmoid activation function. The output of this sigmoid layer is multiplied by the original concatenated features, convoluted, and passed through another fully connected layer with sigmoid activation function. From: [14]

Dual hierarchical attentive aggregations are used to preserve low-level details in the image [65]. The attentive aggregation nodes try to find the shadow regions and their shadow mask and

generate a shadow-free image by concatenating the information from the dilated convolutional layers and previous nodes, with four possible nodes $n = 1, 2, 3, 4$. The structure of the attentive aggregation nodes is shown in Figure 3-5. First, the features from the previous attention node (if available) and dilated convolution layers are concatenated for node n :

$$C_n = C(A_{n-1}, A_{n-2}, L_1^n, L_2^n) \quad (3-8)$$

where $C(\cdot)$ is the concatenation operation, C_n are the concatenated features at node n , and A_{n-1} is the output of the previous attention aggregation node. $A_{n-1} = 0$ if $n < 2$ and $A_{n-2} = 0$ if $n < 3$, $L_1^n = DC_{k=2^{n-1}}$, $L_2^n(x) = DC_{k=2^n}$, and DC_k is the output of the dilated convolution layer with dilation rate k .

C_n is used as input for network $GN_n(\theta_n, \mathbf{x})$, consisting of a pooling layer, two hidden layers with linear activation function, and a hidden layer with sigmoid activation function as explained in Chapter 2. The output of this network is multiplied with the original concatenation and processed with a convolutional layer $GC_n(\theta_{c,n}, \mathbf{x})$ with a 3×3 kernel.

$$A_n = GC_n(\theta_{c,n}, C_n \times GN_n(\theta_n, C_n)) \quad (3-9)$$

where $\theta_{c,n}$ are the weights and biases of the convolutional layer, θ_n are the weights of network GN_n , and A_n is the output of the feature aggregation node n , shown in **3a** in Figure 3-3.

If the node is an attention aggregation node, shown in Figure 3-3, box**3b**, the output is defined as:

$$AT_n = GS_n(\theta_{s,n}, A_n) \quad (3-10)$$

where GS_n is a hidden layer with sigmoid activation function and weights θ_s .

3-2-4 Spatial pooling pyramid

A SPP is used after the last convolutional layer to remove the fixed-size input constraint of the network. A SPP applies max pooling (as in Section 2-3-2) to each of the d feature maps [16] with different window sizes based on the size of the pooling pyramid and creates a vector of the max pooled values, as shown in Figure 3-6.

The level of the SPP is defined as the number of windows with different sizes we pass over the feature map of size $a \times b$. For each level l of the SPP with L levels, the window size $w_1 \times w_2$ is equal to $\frac{a}{l} \times \frac{b}{l}$. The stride used in max pooling is equal to the window size. The max pooling values of all SPP levels are concatenated, with a fixed-length output of size $\sum_1^L (2^{(l-1)})^2$ for each of the d feature maps.

3-2-5 Loss function

The loss function of the DHAN network consists of three parts: a perceptual loss, mask loss (attention loss), and adversarial loss. The perceptual loss is defined as:

$$\mathcal{L}_\Phi = \sum_{k=1}^5 \|\Phi_k(\text{DHAN}(I_s)) - \Phi_k(I_{\text{free}})\|_1 \quad (3-11)$$

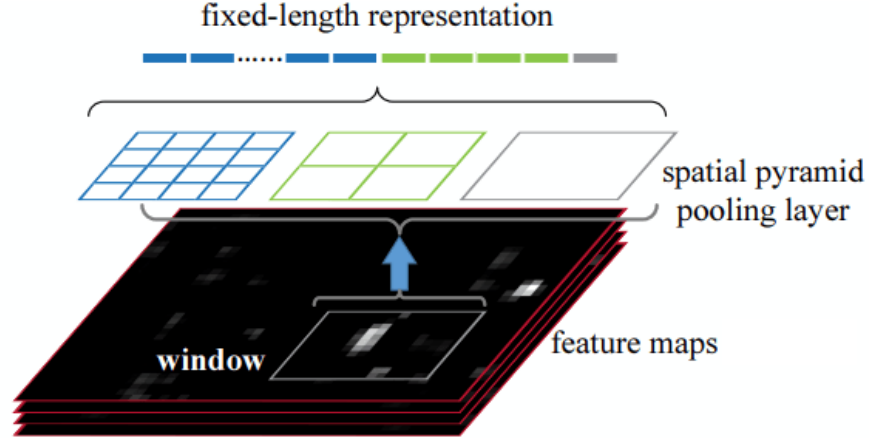


Figure 3-6: Overview of Spatial Pooling Pyramid (SPP). The feature maps are max pooled with three different window sizes. The first window is size $a \times b$, generating a 1×1 max pool matrix (gray). The second window is size $\frac{a}{2} \times \frac{b}{2}$, generating a 2×2 max pool matrix (green). The last window is size $\frac{a}{4} \times \frac{b}{4}$, generating a 4×4 max pool matrix (blue). The max pool values are stored as a vector, generating a fixed-length representation of the feature maps of size $21 \times d$. From [16]

where Φ_k are the weights of the VGG16 network up to layer k , and I_f and $\text{DHAN}(I_s)$ are the Ground Truth (GT) shadow-free image and DHAN generated shadow-free image, respectively. The perceptual loss is applied to check the similarity between the GT and DHAN-generated image in the pixel and feature domains [66]. The second loss is the mask loss. The mask loss is based on the binary cross entropy loss between the GT mask M and generated mask M_D :

$$\mathcal{L}_m = - [M \log (M_D) + (1 - M) \log (1 - M_D)] \quad (3-12)$$

Where N is the number of masks. The final loss is the adversarial loss, as explained in Section 2-3-3:

$$\mathcal{L}_{GAN} = \log(D(I_s, G(I_s))) - \log(D(I_s, I_f)) \quad (3-13)$$

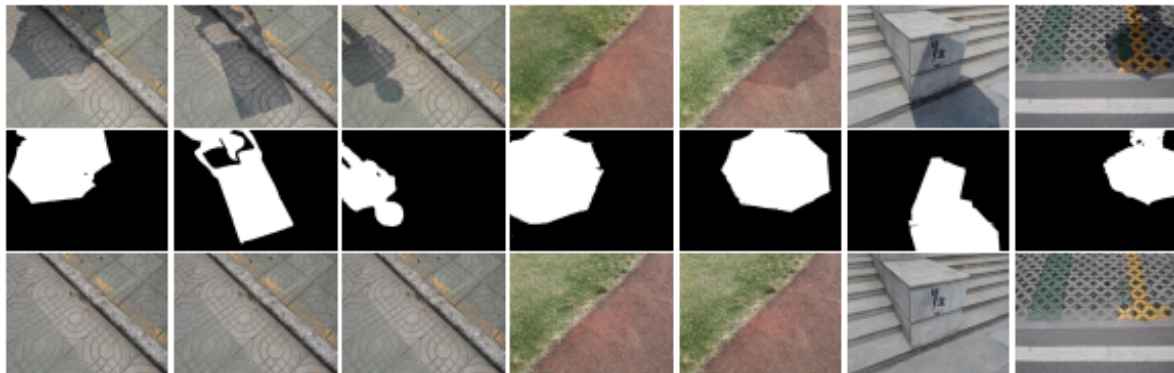
We combine all three loss functions and get $\mathcal{L} = \lambda_l \mathcal{L}_\Phi + \lambda_G \mathcal{L}_{GAN} + \lambda_m \mathcal{L}_m$, with λ_l , λ_G and λ_m the loss weights for the perception, adversarial and mask losses respectively.

3-2-6 Datasets

There are two datasets for shadow removal: the Image Shadow Triplets Dataset (ISTD) and Shadow Removal Dataset (SRD) datasets.

Image Shadow Triplets Dataset

The ISTD dataset consists of image triplets: a shadow image I_s , a mask of the shadow M and shadow free GT I_f [67]. The total amount of triplets is 1870 with 135 scenes. An example of the shadow images is shown in Figure 3-7a.



(a) ISRD shadow triplets



(b) SRD shadow pairs

Figure 3-7: (a) Example triplets from the ISTD dataset and (b) example pairs from the SRD dataset.

Shadow Removal Dataset

The SRD dataset consists of shadow and shadow-free image pairs. The dataset includes various shapes, geometries, sizes and perspectives of shadows. In total it includes 3088 images [68]. An example of the SRD image pairs is shown in Figure 3-7b. The ISTD and SRD dataset are used to train and test both the DHAN as Shadow Generative Adversarial Network explained in the next section.

3-3 Shadow Generative Adversarial Network

The Shadow Generative Adversarial Network (SGAN) is an unsupervised shadow removal method. An unsupervised method is a Deep Learning (DL) method that does not depend on paired data x_i, y_i , as opposed to a supervised method that needs labelled pairs x_i, y_i . In contrast to DHAN, it does thus not need paired input-output data and can be trained to remove shadows based on two datasets: a dataset with images containing shadows and a shadow-free dataset [18]. In this section, we will discuss the SGAN differences from the GAN. Then, the loss function of the network and cycle consistency is explained.

3-3-1 Dual generator and discriminator network

The SGAN has the same working as a standard GAN explained in Section 2-3-3, except that it uses two generators and discriminators. The idea of two generators is inspired by the Cy-

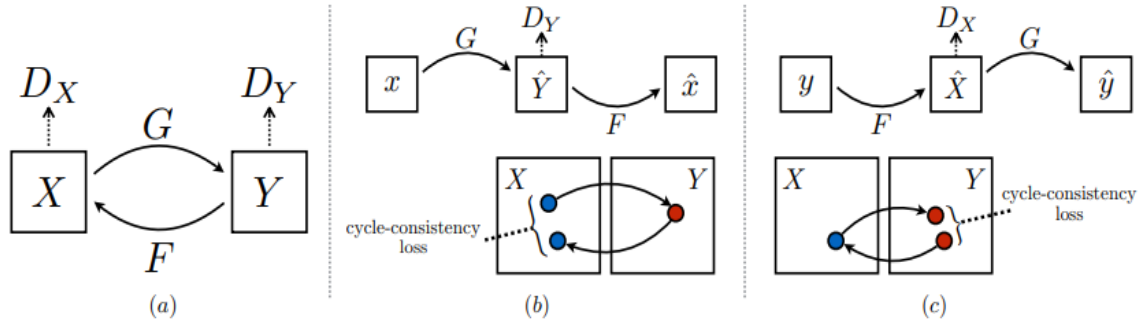


Figure 3-8: A dual generator-discriminator network with cycle consistency. In **(a)**, the generators G and F are shown, where $G : X \rightarrow Y$ and $F : Y \rightarrow X$ and their accompanying discriminators D_X and D_Y respectively. D_Y tries to distinguish fake images generated by G from images in dataset Y and D_X tries to distinguish fake images generated by F from images in dataset X . In **(b)**, the cycle consistency criterion is shown. The intuition is that an image mapped from X to Y by G , F should be able to map it back into its original domain. This is called the forward cycle consistency. In **(c)**, we see the same as in **(b)**, but for mappings from Y to X . This is called the backward cycle consistency. From: [17]

cleGAN principle, where two datasets were available: dataset X with containing images with horses and dataset Y , containing images with zebras [17]. The idea is to build two generators, one generator that can map images from X to Y , and a generator able to map images from Y to X , and two discriminators that can whether the image is one of the real images from the data or a generated fake image. This process is shown in Figure 3-8a. Both generator and discriminator networks work the same as the GAN explained in Section 2-3-3.

The importance of the two generator-discriminator network is the ability to include cycle properties. Since there is no ground truth to train against, the intuition that if we translate an image from one domain to another, we should also be able to translate it back. The forward and backward cycle consistency as shown in Figure 3-8 are defined as $F(G(x)) \approx x$ and $G(F(y)) \approx y$ respectively. The SGAN utilizes this cycle consistency for mapping shadow to shadow-free and vice versa by adding the cycle consistency loss to its loss function. We define the shadow to shadow-free generator as G_f and the shadow-free to shadow generator as G_s with accompanying discriminators D_f and D_s .

3-3-2 Loss function

The SGAN loss function is build from three loss pairs: the cycle-consistency loss, the identity loss and the adversarial loss. The losses are illustrated in Figure 3-9. By utilizing losses from both the shadow and shadowfree GAN processes, we try to utilize the transitivity of the data to be able to translate from one domain to another without the need of paired data [69]. First, we define the cycle consistency loss for both the shadow and shadowfree situation:

$$\mathcal{L}_{cycle}^{(a)}(G_f, G_s) = \|G_s(G_f(I_s)) - I_s\|_1 \quad (3-14)$$

$$\mathcal{L}_{cycle}^{(b)}(G_s, G_f) = \|G_f(G_s(I_f), M_r) - I_f\|_1 \quad (3-15)$$

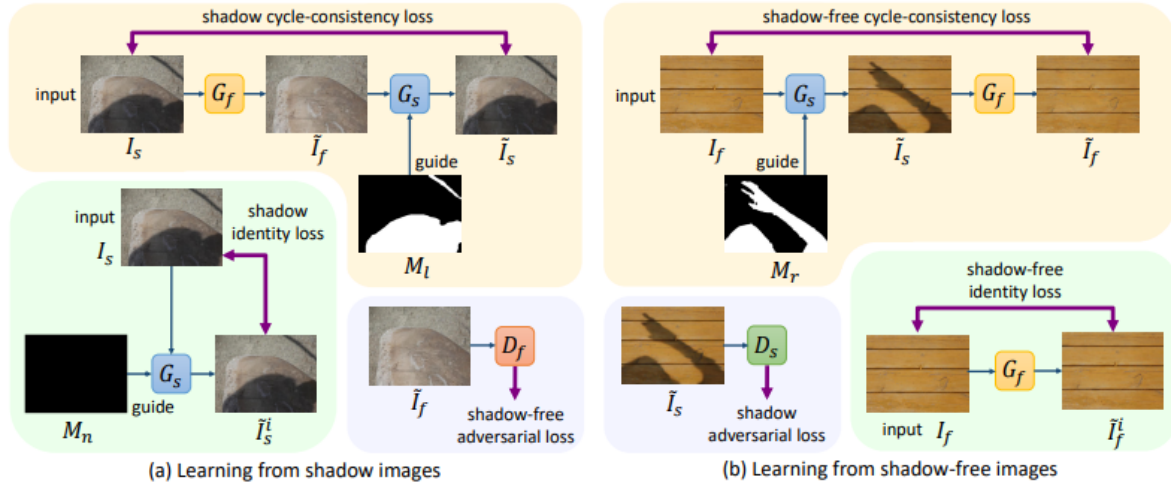


Figure 3-9: The SGAN loss pairs. In (a) we see the shadow cycle consistency loss (yellow), the shadow identity loss and adversarial (discriminator) loss of the images generated by G_f (skyblue). In (b), the shadow-free cycle consistency loss (yellow), shadow adversarial (discriminator) loss (skyblue) and shadow-free identity loss (green) are shown. From: [18]

where \mathcal{L}^a and \mathcal{L}^b denote the shadow and shadow-free loss respectively, as depicted in Figure 3-9. The benefit of a consistency loss has been explained in the previous section. The mask M_r is a guide mask generated when removing shadow. M_r is a binary map, where non-shadow pixels are zeros (depicted in black) and shadow area pixels are one (depicted in white) calculated according to:

$$M = \mathbb{B}(G_f(I_s) - I_s, t) \quad (3-16)$$

where \mathbb{B} is the operator creating a binary image with threshold t using Otsu's threshold algorithm [70]. Next, we define the adversarial loss as:

$$\begin{aligned} \mathcal{L}_{GAN}^a(G_f, D_f) &= \mathbb{E}_{I_f \sim p_{\text{data}}(I_f)} [\log(D_f(I_f))] \\ &+ \mathbb{E}_{I_s \sim p_{\text{data}}(I_s)} [\log(1 - D_f(G_f(I_s)))] \end{aligned} \quad (3-17)$$

$$\begin{aligned} \mathcal{L}_{GAN}^b(G_s, D_s) &= \mathbb{E}_{I_s \sim p_{\text{data}}(I_s)} [\log(D_s(I_s))] \\ &+ \mathbb{E}_{I_f \sim p_{\text{data}}(I_f)} [\log(1 - D_s(G_s(I_f, M_r)))] \end{aligned} \quad (3-18)$$

where p_{data} denotes the data distribution of shadow-free data I_f or shadow data I_s .

Finally, the identity loss is defined as:

$$\mathcal{L}_{\text{identity}}^{(a)}(G_s) = \|G_s(I_s, M_n) - I_s\|_1 \quad (3-19)$$

$$\mathcal{L}_{\text{identity}}^{(b)}(G_f) = \|G_f(I_f) - I_f\|_1 \quad (3-20)$$

where the mask M_n is a binary image without a mask present (a completely black image) as shown in the green region of Figure 3-9a. The identity loss constraints the generators to only remove the shadow or add shadow in the mask defined locations. Furthermore, it enforces the generator G_f not to make alterations to an image that is from dataset Y and generator

G_s not to make alterations to an image that is from dataset X . It follows the intuition that G_f should not alter the image if it is fed an image without shadow and G_s should not add shadow if it is already present.

If we combine the three loss pairs, we find the following loss function for the SGAN network:

$$\begin{aligned} \mathcal{L}_{\text{final}}(G_s, G_f, D_s, D_f) = & \lambda_1 \left(\mathcal{L}_{GAN}^a(G_f, D_f) + \mathcal{L}_{GAN}^b(G_s, D_s) \right) \\ & + \lambda_2 \left(\mathcal{L}_{\text{cycle}}^a(G_f, G_s) + \mathcal{L}_{\text{cycle}}^b(G_s, G_f) \right) + \lambda_3 \left(\mathcal{L}_{\text{identity}}^a(G_s) + \mathcal{L}_{\text{identity}}^b(G_f) \right) \end{aligned} \quad (3-21)$$

where λ_1 , λ_2 and λ_3 are the weights applied to each part of the loss function. The complete loss function is optimized in a min max fashion as in Section 2-3-3:

$$\arg \min_{G_s, G_f} \max_{D_s, D_f} \mathcal{L}_{\text{final}}(G_s, G_f, D_s, D_f) \quad (3-22)$$

The SGAN is trained and tested using the same datasets as the DHAN.

Methodology

The goal of this thesis is to evaluate the segmentation performance of Segmentation Network (SegNet) after shadow removal. This chapter discusses the training setup of the networks in Chapter 3. Then, the metrics to evaluate shadow removal and segmentation performance are discussed. This includes a mathematical overview of the methods. Finally, the experiments are described, and possible alterations to the datasets are stated.

4-1 Training setup

Training the networks described in Chapter 3 is done according to the specifics described in this section. For each of the networks, the SegNet, Dual Hierarchical Aggregation Network (DHAN), and Shadow Generative Adversarial Network (SGAN), the training variables and method are discussed.

SegNet

The SegNet network discussed in Section 3-1 is trained using the CamVid dataset. To initialize the training of the network, the initial weights have to be set. The common method is to initialize the weights with random numbers drawn from a Gaussian distribution with fixed standard deviation [71]. The SegNet network is initialized with some small changes to this process. The biases are initially set to zero, and instead of a fixed variance, the variance is set according to the constraint $\frac{1}{2}n_{i,j} \text{Var}(\mathbf{w}_{i,j}) = \mathbf{1}$, with n the number of connections of neuron (i, j) [72].

The optimization strategy to optimize the weights is Stochastic Gradient Descent (SGD) with fixed learning rate $\eta = 0.1$ and momentum $\gamma = 0.9$. Before each epoch, the training set is shuffled, and images are picked in order. This ensures that each image is only utilized once per epoch, but different images are used for the initialization.

Some classes are represented more (in pixels) than others. To train the network equally over all classes, class balancing is applied. The method used for class balancing is the median frequency balancing method [73]. Median frequency balancing is defined as:

$$a_c = \frac{\text{mFreq}}{\text{freq}_c}$$

where:

$$\text{freq}_c = \frac{\sum_{i=1}^I \text{pixels}_c(c) \ln(i)}{\sum I_{\text{containing class}(c)} \times \text{size}(I)}$$

and:

$$\text{mFreq} = \frac{\sum_{c=1}^N \text{freq}_c}{N}$$

Where N is the number of images, and c is the class. a_c is the class weight of class c . The implication is that the classes with a large presence in the data have a value $a_c < 1$, and the smaller classes will have a balancing class weight $a_c > 1$. This is to prevent overfitting on the more represented classes.

Dual Hierarchical Aggregation Network

The DHAN network weights are initialized by picking from a Gaussian distribution with a standard deviation of 0.01. The optimization method is the SGD. The learning rate is set to 0.002 and 0.001 for the generator and discriminator, respectively. The learning rate decays towards zero starting from epoch 100. The loss weights λ_l , λ_Φ , and λ_m are set to 10, 60 and 20 respectively.

Shadow Generative Adversarial Network

The SGAN network is initialized by picking starting weights from a Gaussian distribution with a standard deviation of 0.02 [71]. The optimization method is SGD with momentum. The learning rate is set at 0.002 and momentum at 0.5 for the first 100 epochs. From epoch 100 to 200, the learning rate decays from 0.002 to zero, and the momentum increases from 0.5 to 0.999. This holds for both generators and discriminators. The loss function weights λ_1 , λ_2 and λ_3 are set to 1, 10 and 5 respectively [18]. The loss function is discussed in Section 3-3.

Hardware

The GPU used for training was the NVIDIA RTX 3090. The CPU in the system was the Intel Core i9-10900X with 128GBI RAM.

4-2 Evaluation metrics

In this section, we describe three methods to assess the performance of shadow removal and segmentation. The Structural Similarity Index Measure (SSIM) metric is used for the analysis of shadow removal, and the Pixel Accuracy (PA) and mean Intersection over Union (mIoU) metrics are for segmentation performance.

4-2-1 Structural Similarity Index Metric

The structural similarity measures the perceived change in structural information of two images. The structural similarity is calculated per pixel by moving a Gaussian 7×7 window over image X and Y and calculating the mean and variance. The SSIM is calculated per window (\mathbf{x}, \mathbf{y}) . The means μ_x , μ_y , and variances σ_x , σ_y are used to compute the structural similarity of the window. The SSIM consists of three components: luminance, contrast, and structure.

The luminance component of the structural similarity is defined by:

$$l(\mathbf{x}, \mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \quad (4-1)$$

where $C_1 = (255K_1)^2$ with $K_1 \ll 1$ is a small constant to enforce numerical stability if $\mu_x^2 + \mu_y^2$ gets close to zero. The luminance component is qualitatively consistent with Weber's law, which has been used to model light adaptation [74].

The contrast component of the SSIM is defined as:

$$c(\mathbf{x}, \mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (4-2)$$

Where $C_2 = (255K_2)^2$ with $K_2 \ll 1$ is again a small constant to enforce numerical stability. The structure component is defined as:

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \quad (4-3)$$

Again, $C_3(255K_3)^2$ with $K_3 \ll 1$ is a constant to enforce numerical stability and σ_{xy} the correlation coefficient defined as:

$$\sigma_{xy} = \frac{1}{N-1} \sum_{i=1}^N (x_i - \mu_x)(y_i - \mu_y) \quad (4-4)$$

The total structural similarity is then a sum of its components:

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = [l(\mathbf{x}, \mathbf{y})]^\alpha \cdot [c(\mathbf{x}, \mathbf{y})]^\beta \cdot [s(\mathbf{x}, \mathbf{y})]^\gamma \quad (4-5)$$

During this thesis, the weights α , β and γ have been set to 1 and we define $C_3 = \frac{C_2}{2}$. This gives us the following form:

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4-6)$$

The output of the SSIM window is combined into an image of the same size as the original input images. We define the final output as $I_{\text{ssim}} = \text{SSIM}(X, Y)$, where each index (i, j) of I_{ssim} contains a value between zero and one, zero stating there is no structural similarity, and one meaning perfect structural similarity. $I_{\text{ssim}}(i, j)$ corresponds to the structural

similarity between the input images X and Y around pixel (i, j) . The SSIM value of each index (i, j) states how much that pixel in image X is structurally similar to the same pixel in Y .

During this thesis, when speaking of the total structural similarity of an image, we speak of the mean structural similarity of that image defined as the sum of all pixel structural similarities:

$$\text{mSSIM}(X, Y) = \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M I_{\text{ssim}}(i, j) \quad (4-7)$$

Where a mSSIM of zero states that there is no structural similarity between the images, and one means perfect structural similarity (identical images).

Compared to metrics such as the Mean Squared Error (MSE), the SSIM does not use the difference between two pixels but looks for similarities. The intuition behind this is that it tries to determine if the pixels in the two images line up and/or have similar pixel density values [14, 74].

4-2-2 Pixel Accuracy

The global pixel accuracy is the number of pixels classified correctly by the network divided by the total number of pixels in an evaluated image [33].

$$\text{PA} = \frac{\sum_{j=1}^k n_{jj}}{\sum_{j=1}^k t_j}, \quad \text{with } k = 0, 1, \dots, K \quad (4-8)$$

Where n_{jj} is the number of pixels both labelled and classified as class j and t_j is the total amount of pixels. This accuracy measure can also be used class-wise and is called the mean pixel accuracy. The mean pixel accuracy calculates the accuracy per class k and takes the mean of the accuracy over the total amount of classes:

$$\text{mPA} = \frac{1}{k} \sum_{j=1}^k \frac{n_{jj}}{t_j} \quad (4-9)$$

Where k is the amount of classes, n_{jj} is the number of pixels labelled and classified as class j and t_j is the total number of pixels.

The global and mean pixel accuracy are common metrics to evaluate the performance of a segmentation network [9].

4-2-3 Intersection over Union

The Intersection over Union (IoU) metric can be used to assess the performance of a segmentation network. The IoU metric is also called the *Jaccard index* [75]. It measures the similarity between two finite areas by dividing the area of overlap between the Ground Truth (GT) A and prediction B by the total area of the GT and prediction:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (4-10)$$

This states that the IoU overlap between the ground truth and prediction is divided by the total area of both the ground truth and prediction. As with the PA, the IoU metric can be adjusted to a class-based variant. The mIoU is the IoU calculated per class k , averaged for the total number of classes K .

$$\text{mIoU} = \frac{1}{K} \times \sum_{k=1}^K \text{IoU}_k \quad (4-11)$$

The mIoU is the standard evaluation metric for object detection, and segmentation [13].

4-3 Experiments

In this section, the experiments conducted during this thesis are stated. The first experiment describes the shadow removal process using the DHAN and SGAN shadow removal networks. It evaluates the networks and datasets in-depth, to make an educated decision for a preferred shadow removal method. The evaluation metric used is the SSIM. Then, we segment the shadow-removed images and benchmark the results against the images segmented without shadow removal. Inspired by the results of these experiments, another set of experiments is proposed. This set of experiments tries to generate a dataset containing shadows that more closely resembles the CamVid dataset. After the generation of a synthetic dataset, the same evaluation steps are taken as in the first experiment.

4-3-1 Comparison of removal methods

For the first experiment, DHAN and SGAN shadow removal networks are evaluated on both the Image Shadow Triplets Dataset (ISTD) and Shadow Removal Dataset (SRD) datasets, with a 200 epoch training time. As explained in the previous chapter, the DHAN and SGAN differ in their approach and learning type, comparing a supervised method (DHAN) with an unsupervised method (SGAN).

After training of the networks on both datasets, we evaluate the effects of removal by comparing them to the GT using the SSIM. In most papers, the average structural similarity over the whole dataset is shown [14, 18]. Instead, we compare the SSIM by calculating its distribution.

In a second experiment with the structural similarity, the separate components of the SSIM are compared in non-shadow regions to test if the networks alter pixels in non-shadow regions. To compare the non-shadow region structural similarity of the images, we create inverted mask images. Inverting the masks means that the pixel value of pixels located in a shadow region defined by the mask M will be set to zero. To create an inverted mask imageset I' , we can use the following equation:

$$I^{(k)'}(i, j) = \begin{cases} I^{(k)}(i, j) & \text{if } M^{(k)}(i, j) = 0 \\ 0 & \text{if } M^{(k)}(i, j) = 1 \end{cases} \quad \text{with } i = 1, 2, \dots, w \quad j = 1, 2, \dots, h \quad (4-12)$$

where I is the original image, M is the shadow mask, w is the width of the image, h is the height of the image and $k = 1, 2, \dots, N$ is the image index from the image set I with N images. The image set with inverted masks will then be $I' = [I'_1, I'_2, \dots, I'_N]$. The Input, GT, DHAN

output and SGAN output image sets after inverting the mask are defined as I'_s , I'_f , I'_{DHAN} and I'_{SGAN} respectively.

4-3-2 Segmentation experiments

In this experiment, we quantify the results of segmentation after applying shadow removal to the CamVid dataset and make a visual inspection of the results. This visual inspection leads to an analysis of the brightness of the segmented images using the intensity difference between the CamVid image after and before shadow removal:

$$I_{\text{diff}}^{(k)} = \text{DHAN}(I_c^{(k)}) - I_c^{(k)} \quad (4-13)$$

where I_c^k is image k from the CamVid dataset. The complete imageset I_{diff} is defined as $I_{\text{diff}} = \{I_{\text{diff}}^{(1)}, I_{\text{diff}}^{(2)}, \dots, I_{\text{diff}}^{(N)}\}$ with N the size of the dataset. The second part of this experiment consist of a shadow analysis, where we evaluate the shadows from the ISTD and SRD datasets to create a synthetic shadow dataset.

The goal of the synthetic dataset is to create a dataset for training the DHAN and testing the effect on semantic segmentation accuracy. The synthetic shadow datasets are generated based on the shadow analysis of the ISTD and SRD datasets. The shadow intensity difference, defined as the difference between the shadow-free and shadow images $I_{\text{diffs}} = I_f - I_s$, is calculated to define the pixel intensity difference between the shadow and shadow-free images. Due to pixel value deviations in non-shadow areas, shown in the shadow analysis experiments defined in Section 4-3-1, the intensity difference is set to zero in non-shadow regions according to:

$$I_{\text{diffs}}^{(k)}(i, j) = \begin{cases} I_f^{(k)}(i, j) - I_s^{(k)}(i, j) & \text{if } M^{(k)}(i, j) = 1 \\ 0 & \text{if } M^{(k)}(i, j) = 0 \end{cases} \quad \text{with } i = 1, 2, \dots, w \quad j = 1, 2, \dots, h \quad (4-14)$$

with the same parameters as in Equation 4-12. Intuitively, we can describe the intensity difference as the darkness of the shadow. The larger the intensity difference, the darker the shadow.

Based on the distribution of the intensity differences, we generate synthetic shadow datasets, where shadow is added to the CamVid images. To evaluate different shadow intensities, where a higher intensity means a darker shadow. An example of the increasing shadows in images is shown in Appendix B-2. In total, eleven synthetic datasets are generated. We define the synthetic datasets as DataSyn_d , where $d = 1, 2, \dots, 10$. The first eleven datasets have an increasingly darker shadow with $d = 1$ the smallest I_{diffs} and therefore the lightest shadow and $d = 10$ the darkest shadow.

The shadow is added to each image in the CamVid dataset I_c for a set of predefined CamVid

masks M_c according to:

$$\text{DataSyn}_d^{(k)}(i, j, p) = \begin{cases} I_c^{(k)}(i, j, 1) - \lfloor (6.2(d-1) + 50) \rfloor & \text{if } M_c^{(k)}(i, j) = 1 \text{ and } p = 1 \\ I_c^{(k)}(i, j, 2) - \lfloor (5.9(d-1) + 44) \rfloor & \text{if } M_c^{(k)}(i, j) = 1 \text{ and } p = 2 \\ I_c^{(k)}(i, j, 3) - \lfloor (5.6(d-1) + 39) \rfloor & \text{if } M_c^{(k)}(i, j) = 1 \text{ and } p = 3 \\ I_c^{(k)}(i, j, p) & \text{if } M_c^{(k)}(i, j) = 0 \end{cases} \quad (4-15)$$

where $i = 1, 2, \dots, w$, $j = 1, 2, \dots, h$, $p = 1, 2, 3$. k denotes the image within an imageset of size N and d denotes the intensity of the shadow. The final dataset created, DataSyn_{11} , contains the images of all previous datasets:

$\text{DataSyn}_{11} = \{\text{DataSyn}_1, \text{DataSyn}_2, \dots, \text{DataSyn}_{10}\}$. For each of the DataSyn datasets, a DHAN model is trained. The trained models are defined as DHAN_i where i is the DataSyn dataset used for training. We evaluate the performance of each of the generated models on all DataSyn datasets. The metrics to evaluate the segmentation are the PA and mIoU, as explained at the start of this chapter.

Chapter 5

Results

In this chapter, experiments are conducted to test how shadow removal can improve semantic segmentation by Segmentation Network (SegNet). The removal methods evaluated are the supervised Dual Hierarchical Aggregation Network (DHAN) and unsupervised cycle-based Shadow Generative Adversarial Network (SGAN).

In Section 5-1, three shadow removal methods are defined: no shadow removal I_s , DHAN shadow removal $DHAN(I_s)$ and SGAN shadow removal $SGAN(I_s)$. Second, a visual comparison of the results is made. Finally, we assess the structural similarity in non-shadow regions for the three removal methods. The removal methods are compared using the Structural Similarity Index Measure (SSIM) metric explained in Chapter 4.

In Section 5-2, segmentation performance without shadow removal, DHAN shadow removal and SGAN shadow removal is compared. Second, we evaluate the increase in pixel values by the DHAN and SGAN and its effect on segmentation performance. The segmentation performance is measured using the Pixel Accuracy (PA) and mean Intersection over Union (mIoU) metrics, as explained in Chapter 4.

In Section 5-3, the shadow of the Image Shadow Triplets Dataset (ISTD) and Shadow Removal Dataset (SRD) are evaluated to create a synthetic dataset. We retrain the DHAN on the synthetic dataset and Section 5-4, the PA and mIoU improvement ratios are shown after retraining on this new dataset.

5-1 Shadow removal methods

In this section, the DHAN and SGAN are compared using the SSIM metric. The networks are trained on both the ISTD and SRD dataset according to the parameters discussed in Chapter 4. The SSIM is calculated for three situations: the input I_s (compared with the shadow-free Ground Truth (GT) I_f), where no shadow removal is conducted, the shadow removed DHAN

output $\text{DHAN}(I_s)$ and the shadow removed SGAN output $\text{SGAN}(I_s)$. We define the three structural similarities:

- Input structural similarity $\text{SSIM}(I_s, I_f)$
- DHAN structural similarity $\text{SSIM}(\text{DHAN}(I_s), I_f)$
- SGAN structural similarity $\text{SSIM}(\text{SGAN}(I_s), I_f)$.

In Section 5-1-1, the Input, DHAN and SGAN structural similarity is shown for the ISTD and SRD datasets. In Section 5-1-2, a visual comparison is made between the Input, DHAN output, SGAN output and GT. In Section 5-1-3, the SSIM is determined for the removal methods for all non-shadow regions according to the inverted mask process described in Chapter 4.

5-1-1 Structural similarity comparison

To evaluate the shadow removal performance of both the DHAN and SGAN, the distribution of the structural similarities of the shadow removal methods are plotted in Figures 5-1a and 5-1b for the ISTD and SRD datasets respectively.

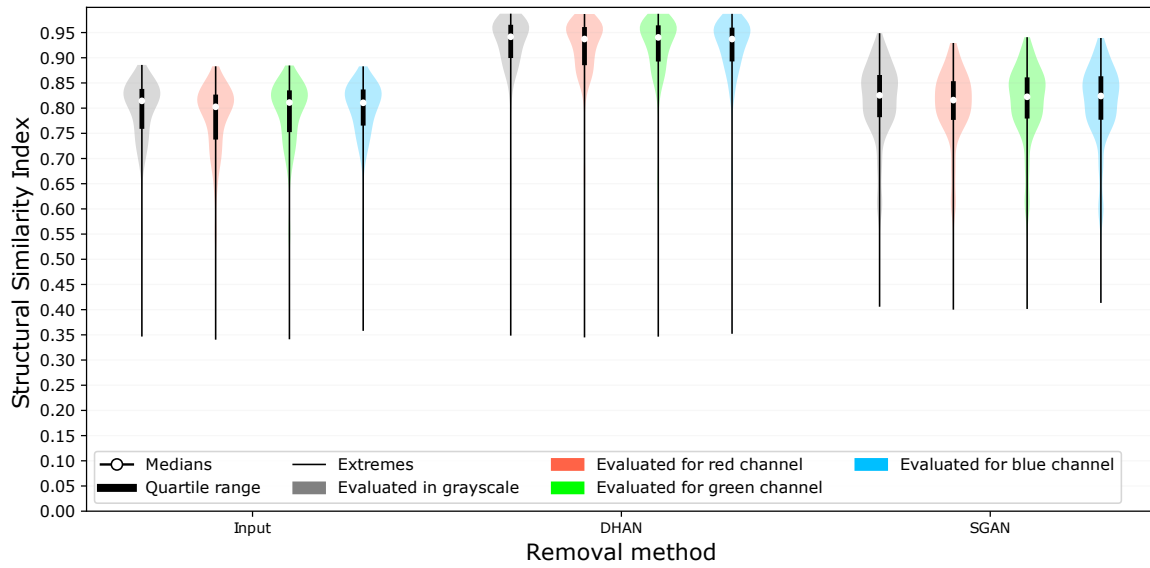
In Figure 5-1a, it is observed that the DHAN structural similarity distribution is located towards higher SSIM scores than the Input or SGAN structural similarities. The DHAN has the highest median (0.94) and maximum (0.99) compared to an Input median of 0.81 and maximum of 0.89. The SGAN shows an improvement compared to the input, with the median at 0.83 and the maximum at 0.95.

In Figure 5-1b, we see similar results. The DHAN structural similarity distribution is located towards the larger SSIM scores, with a median of 0.90 and maximum of 0.98. The SGAN median is located at 0.81 and maximum at 0.95. Compared to the input median (0.74) and maximum (0.88) and SGAN, the DHAN has the distribution located towards the highest SSIM scores.

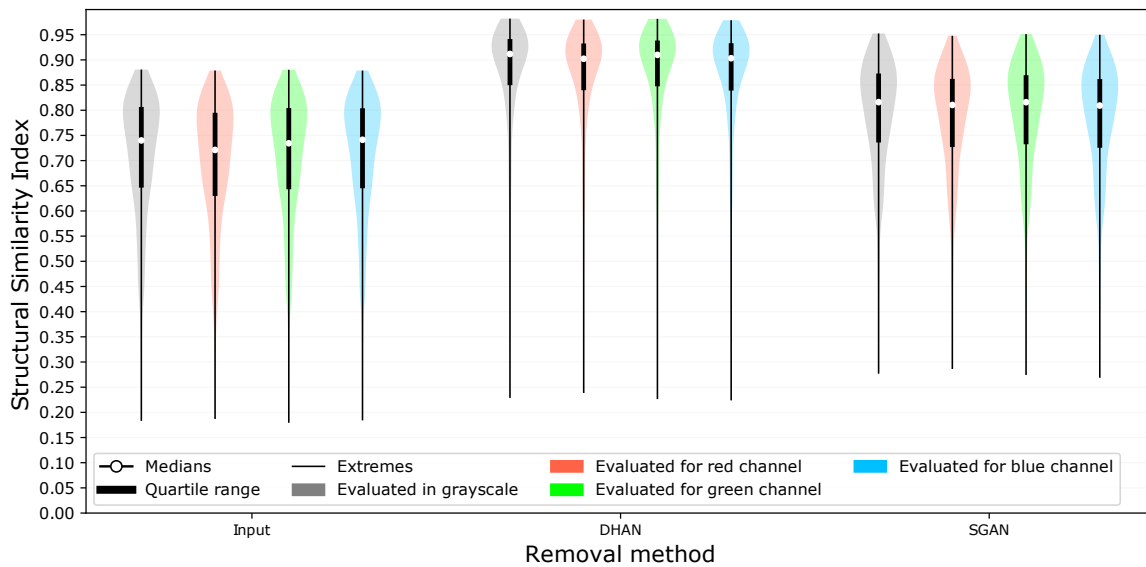
Another observation is that the minima in Figures 5-1a and 5-1b are located at over 5 times Inter Quartile Range (IQR) for all removal methods on both datasets. In many applications, data points further from the quartiles than 1.5IQR are considered as outliers [76]. Figures 5-1a to 5-2b do not give an explanation for these minima.

Based on Figures 5-1a and 5-1b, we can deduct that the DHAN improves the structural similarity if $\text{SSIM}(\text{DHAN}(I_s^k), I_f^k)$ is larger than the maximum structural similarity of the Input. For 81% (ISTD dataset) and 65% (SRD dataset), this holds true. The same evaluation can be made for the SGAN. The SGAN has a structural similarity larger than the maximum of the Input for 13% (ISTD dataset) and 18% (SRD dataset) of the images. This only shows the minimal amount of images that have an increased structural similarity, not the total.

To draw conclusions about the structural similarity improvement per image, the structural



(a) Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the ISTD dataset

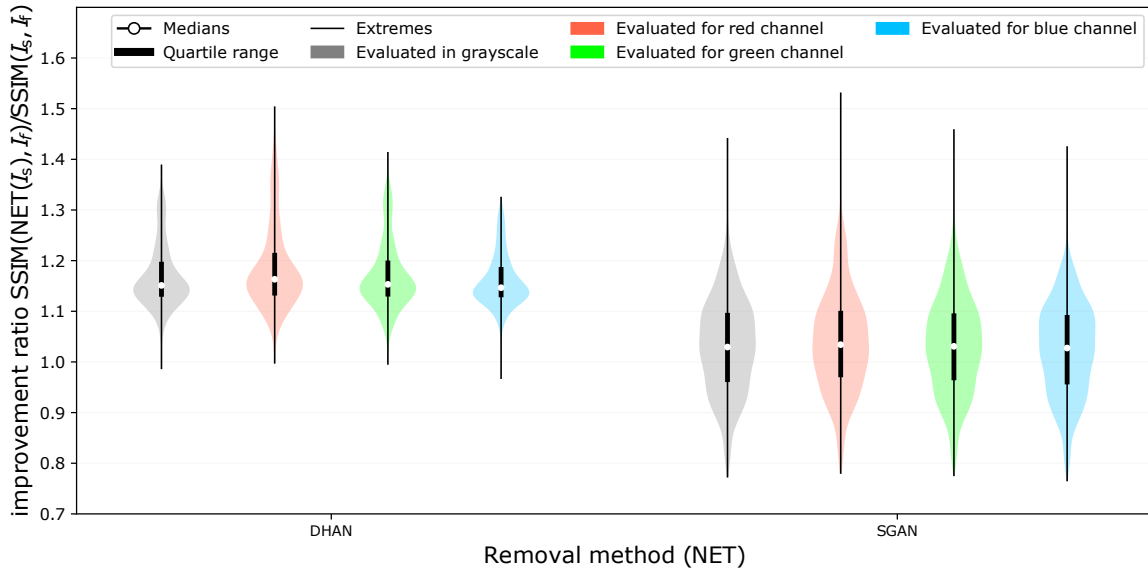


(b) Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the SRD dataset

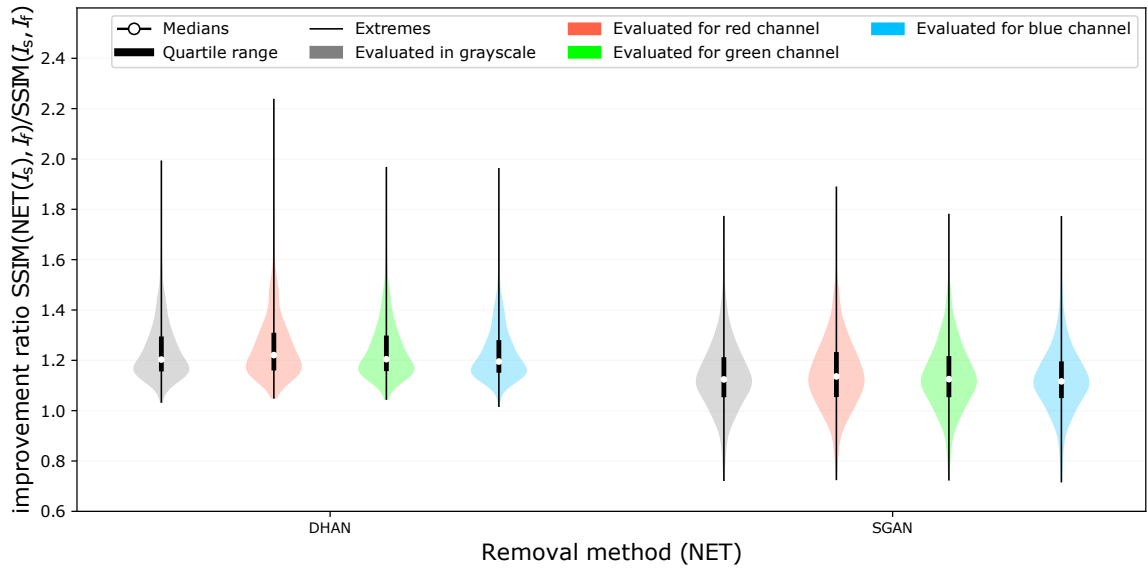
Figure 5-1: Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the (a) ISTD and (b) SRD datasets. For all three removal methods, the structural similarity is shown in grayscale (gray violins) and for each of the three color channels (red-, green-, and blue violins).

similarity improvement ratio between the networks and the input is calculated. The improvement ratio is defined as:

$$\text{SSIM}_{\text{ratio}} = \frac{\text{SSIM}(\text{NET}(I_s), I_f)}{\text{SSIM}(I_s, I_f)} \quad (5-1)$$



(a) Structural similarity improvement ratio between the Input and DHAN- and SGAN for the ISTD dataset



(b) Structural similarity improvement ratio between the Input and DHAN- and SGAN for the SRD dataset

Figure 5-2: Structural similarity improvement ratio between Input and DHAN - and SGAN for the (a) ISTD and (b) SRD datasets. For both networks, the structural similarity in grayscale (gray violins) and for each of the three color channels (red-, green-, and blue violins) is shown.

Where NET is the DHAN or SGAN network, I_s is the Input containing shadows and I_f is the shadow free GT. In Figure 5-2a, the structural similarity improvement ratio is plotted for the ISTD dataset. The DHAN method has minimal $SSIM_{ratio}$ 0.99 on the ISTD dataset. The DHAN increases the structural similarity for almost all images, with an improvement ratio

larger than one for 99.6% of the images. The SGAN improves the structural similarity for 63.3% of the images.

In Figure 5-2b, the improvement ratio for the DHAN and SGAN on the SRD dataset is shown. The smallest improvement ratio of the DHAN on the SRD dataset is 1.01. The DHAN improves the structural similarity for all images of the SRD dataset compared to the Input. The SGAN increases the structural similarity for 85.7% of the images. The DHAN increases the structural similarity for all images.

Based on Figures 5-1a to 5-2b, we can conclude that the DHAN increases the structural similarity for more images than the SGAN, with an increase for 99.6% (ISTD) and 100% (SRD) of the images, compared to 63.3% (ISTD) and 85.7% (SRD). The DHAN increases the structural similarity for more images and in the preferred shadow removal method based on the structural similarity metric.

Color channels

In Figures 5-1a to 5-2b, the SSIM and its improvement ratio are shown in grayscale and for the Red, Green, Blue (RGB) color channels to evaluate the effect of each channel on the structural similarity. Visually, little to no difference can be seen between the distributions in grayscale, red, green or blue. Looking at the means (grayscale: 0.791, red: 0.775, green: 0.787, blue: 0.791) and variance (grayscale: 0.0056, red: 0.0062, green: 0.0057, blue: 0.0052) of the Input for all four color options, we see a maximum difference between the means of 2.0%. For the DHAN, the means (grayscale: 0.922, red: 0.915, green: 0.918, blue: 0.916), the maximal difference between the means is 0.7%. The maximum difference between the means of the SGAN (grayscale: 0.812, red: 0.804, green: 0.809, blue: 0.808) is 1.0%.

Comparing the grayscale to the RGB channel SSIM, the number of images with improved structural similarity remains 99.6% and 100% for the DHAN. The SGAN has the same number of images with an increase in structural similarity for all three color channels compared to grayscale. Due to the same amount of images with improved structural similarity and minimal deviations in the means, we conclude that calculating the structural similarity in grayscale or based on its color channels is statistically indifferent.

Correlation between masksize and structural similarity

The size of the shadow region, also called the masksize, could have an effect on the structural similarity. The effect of the masksize is determined by plotting the masksize against the structural similarity for each of the three removal methods. The results are shown in Figure 5-3 for both the ISTD and SRD datasets.

First, the results on the ISTD dataset are evaluated (top row). Based on the interpretation of correlation coefficients in [77], the correlation between the Input and the masksize (-0.306) is negative and weak. For the DHAN and SGAN, the correlation between the structural similarity and the masksize is negative and negligible (-0.057, -0.087, respectively). The SRD dataset shows a moderate negative correlation for the Input and masksize, whose the

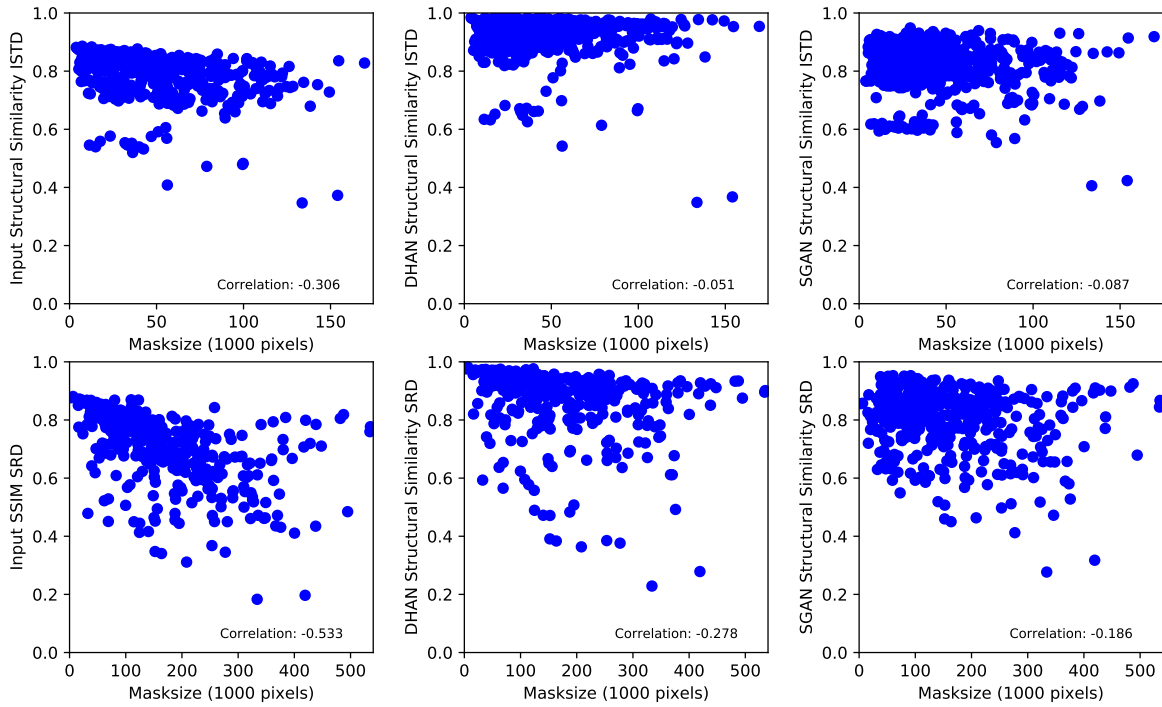


Figure 5-3: The correlation between the masksize and structural similarities of the Input, DHAN and SGAN. The top row shows the results on the ISTD dataset. The bottom row shows the results on the SRD dataset.

DHAN and SGAN show a weak correlation. The weak and moderate correlation between the Input and masksize means that the size of the mask does influence the structural similarity, but not to a large extent. The negligible and weak correlations for the DHAN and SGAN could be explained by the fact that both methods try to remove the shadow from the images. The structural similarity will increase, while the masksize does not change.

5-1-2 Visual performance and error maps

In Figures 5-4 and 5-5, we see the best (+) and worst (-) SGAN performance and best (+) second to worst (-) DHAN performance as rows. The second worst DHAN performance is shown because the SGAN and DHAN have the lowest SSIM score on the same image. For the best-performing images SSIM map, we can see that the structural similarity for the DHAN on locations where the shadow used to be, has increased compared to the Shadow SSIM map. Evaluating the SSIM map of the SGAN, we can see that outside of the shadow mask area, the structural similarity decreases. We hypothesise that this is due to alterations that the SGAN makes to pixels that are located outside of the shadow region.

For the worst performing images, we see that the mask area of the SSIM map is affected but no complete removal is achieved. A remarkable observation is that the structural dissimilarity between the input image and Ground Truth is not limited to the area containing the shadow. It is expected that the Input and GT image are the same in non-shadow regions due to the definition of the input and GT: the GT is the same image as the Input except

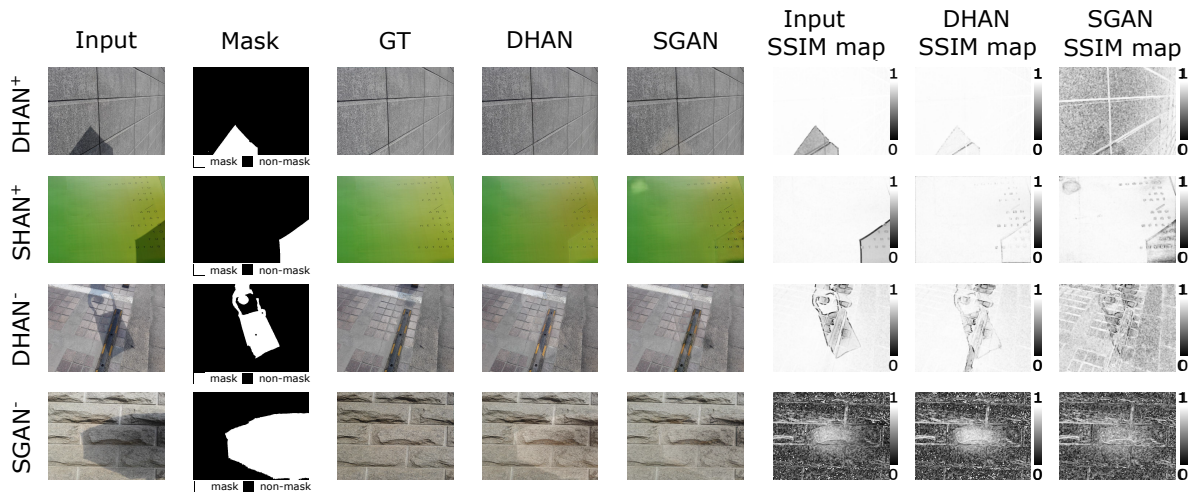


Figure 5-4: Best DHAN and SGAN (DHAN⁺, SGAN⁺) and second worst DHAN (DHAN⁻) and worst (SGAN⁻) performances on the ISTD dataset. Input shows the original image containing a shadow to be removed. Mask contains the binary (black and white) mask of the shadow to be removed. The GT is the target shadow-free image. DHAN and SGAN show the results of the DHAN and SGAN networks on the Input. The Shadow SSIM map, DHAN SSIM map, and SGAN SSIM map show the structural similarity per pixel for the input, DHAN output, and SGAN output with respect to the Ground Truth. The maps show the structural similarity range from black (structural similarity = 0) to white (structural similarity = 1).

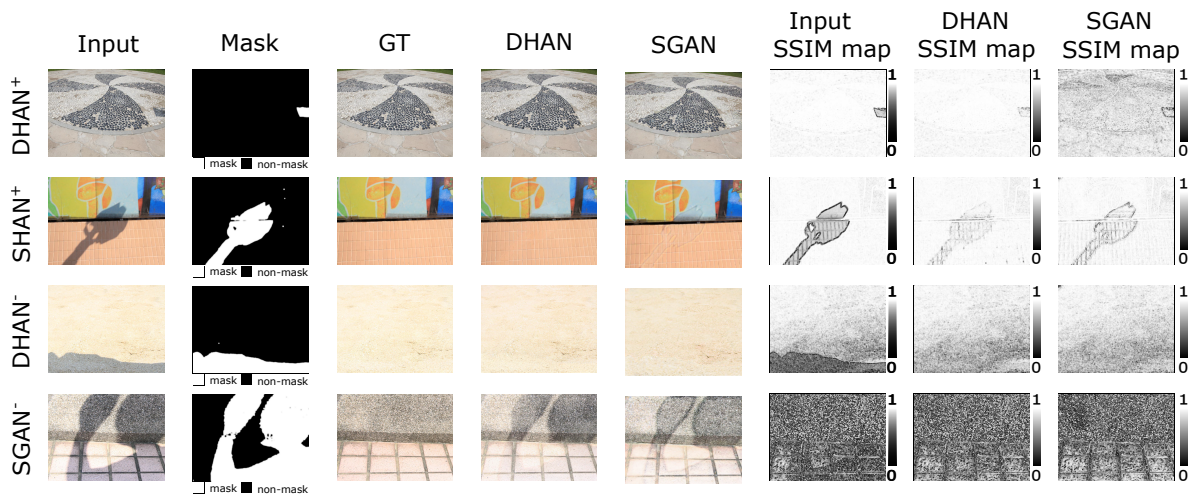


Figure 5-5: The best (+) DHAN and SGAN, second worst DHAN (DHAN⁻) and worst (SGAN⁻) performances on the SRD dataset. Input shows the original image containing a shadow to be removed. Mask contains the binary (black and white) mask of the shadow to be removed. The GT is the target shadow-free image. DHAN and SGAN show the results of the DHAN and SGAN networks on the Input. The shadow SSIM map, DHAN SSIM map, and SGAN SSIM map show the structural similarity per pixel for the input, DHAN output, and SGAN output with respect to the Ground Truth. The maps show the structural similarity range from black (structural similarity = 0) to white (structural similarity = 1).

for the shadow region. This means that the structural similarity between the input image and Ground Truth should be one in regions where there is no shadow. The non-equality

of non-shadow regions between the Input and GT is confirmed by evaluating the SSIM maps of the Input. In non-shadow regions, the SSIM is lower than 1, where a similarity of 1 is expected. To evaluate the possibility of an error in the dataset regarding differences between the input and Ground Truth images outside the shadow region and check our hypothesis that the SGAN decreases the structural similarity outside of the shadow region, the difference between the non-shadow is evaluated.

5-1-3 Image comparison in non-shadow regions

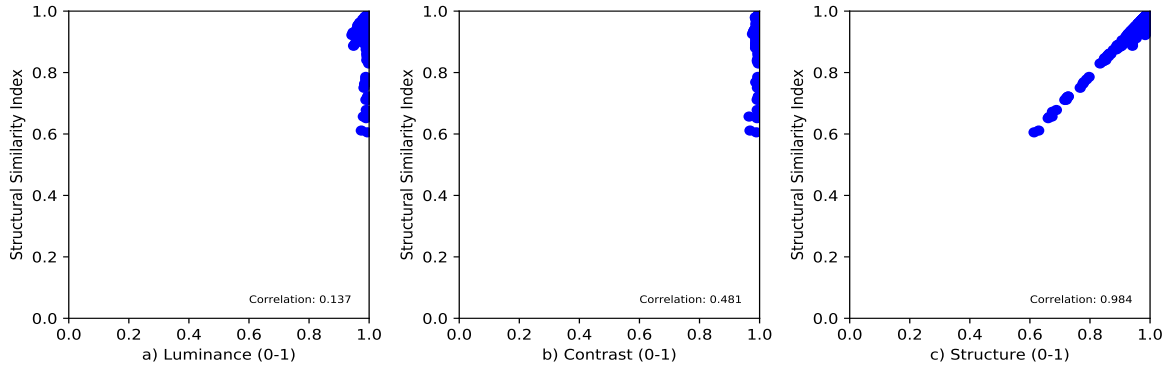
In this section, the ISTD and SRD dataset SSIM values are evaluated with inverted masks to check for dissimilarities between the Input and GT in non shadow regions. Furthermore, we check if the SGAN or DHAN networks change pixel values outside of the shadow region, as hypothesized in the previous section. The non-shadow regions are evaluated by inverting the mask as explained in Section 4-3-1. Setting the pixel values of shadow-region pixels to zero forces the structural similarity in these regions to one. This allows for comparison of the non-shadow regions of the image for the Input, DHAN and SGAN. The three components of the SSIM metric as explained in Chapter 4 are evaluated separately during this section, to find which component causes SSIM deviations. Since the SSIM is a sum of its three components, the luminance, contrast and structure components are also expected to be one.

Input

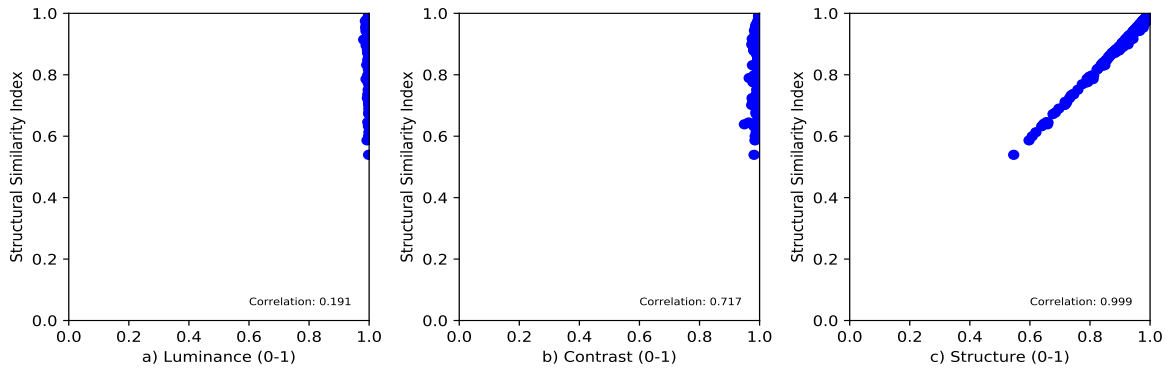
In Figure 5-6, the SSIM components are plotted against the SSIM. First, we observe that the structural similarity between I'_s and I'_f is not equal to one as expected. To investigate the cause of the lower SSIM scores, the correlation between each of the components and the total SSIM is calculated. The correlation between luminance and SSIM (0.137), and contrast (0.418) and SSIM is weak [77] for the ISTD dataset. The structure component, however, does have a very strong correlation (0.984) with the SSIM. For the SRD dataset, the luminance has a weak correlation (0.191), the contrast (0.717) a strong correlation and the structure (0.999) a very strong correlation [77]. Based on these correlations, we state that the luminance and contrast are not the cause of the decrease in structural similarity, but that the structure component causes the decrease in structural similarity between the Input and GT in non-shadow regions.

Second, we conclude that the structural similarity difference between the Input and GT means that subset of images in the dataset have a structural dissimilarity where none is expected. For the images that contain this dissimilarity, no SSIM predictions can be made without containing an error that is inherited from this mismatch between Input and GT non-shadow regions.

In Appendix B-2, the SSIM distributions of Figures 5-1a and 5-1b are shown for the images in the datasets that have an $SSIM(I_s^{(k)}, I_f^{(k)}) > 0.90$. In this Figure, we see that, for the same IQR value, the minimum of over five times the IQR decreases to 1.3 times the IQR for the ISTD network and 2.1 times the IQR for the SRD dataset. We conclude that the existing error in the dataset explains the over five times IQR of the minima in SSIM seen in Section 5-1-1, Figures 5-1a and 5-1b.



(a) Input structural similarity versus its components on the ISTD dataset



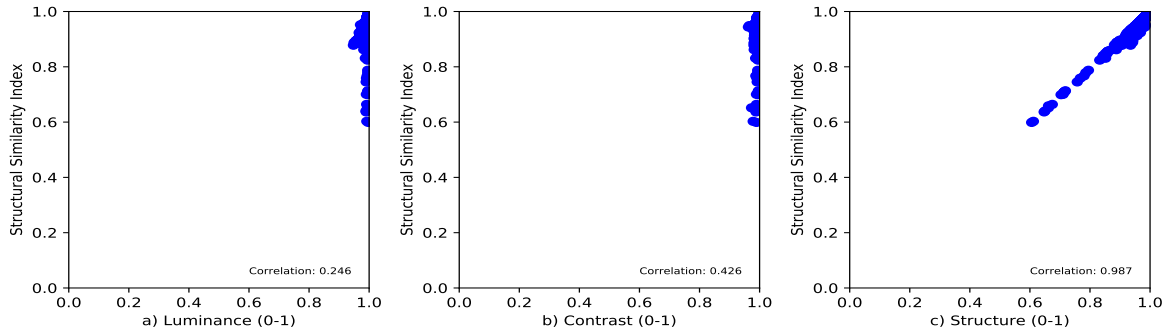
(b) Input Structural similarity versus its components on the SRD dataset

Figure 5-6: $SSIM(I'_s, I'_f)$ for the ISTD (a) and SRD (b) datasets. The SSIM metric is split into its three components; the (a) Luminance, (b) Contrast and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.

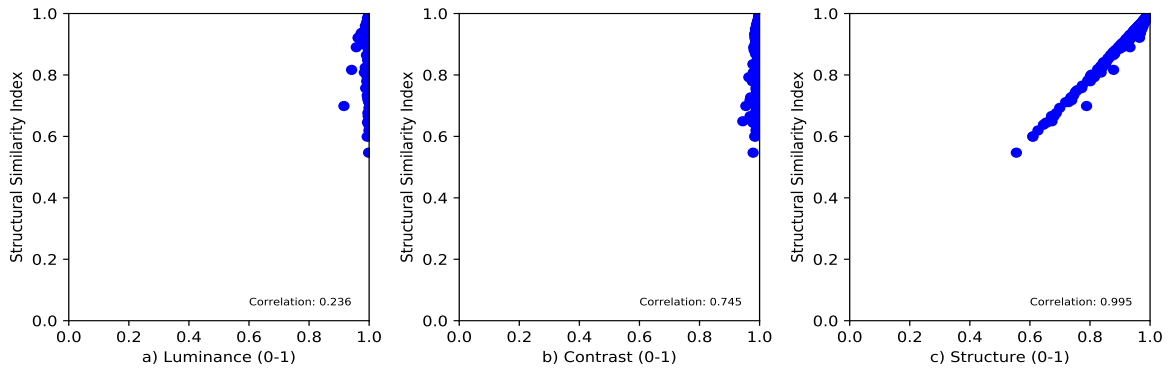
Networks

In Figures 5-7a and 5-7b, the components of the SSIM are plotted against the SSIM of the DHAN inverted mask images I'_{DHAN} . Similar observations as for I'_s are made: The luminance (0.246) and contrast (0.426) elements have weak correlations with the SSIM. The structure (0.987) strongly correlates with the SSIM for the ISTD dataset. For the SRD dataset, the luminance (0.236) has a weak correlation, the contrast (0.745) is strongly correlated, and the structure is very strongly correlated (0.995). The correlations are comparable to the correlations of the Input SSIM with its components for both datasets. Based on these correlations, we hypothesize that the structural similarity is not one because of the dissimilarity between I'_s and I'_f . A second possibility is that the DHAN alters pixel values in non-shadow regions of the image. To check this, the correlation between $SSIM(I'_s, I'_f)$ and $SSIM(I'_{\text{DHAN}}, I'_f)$ is evaluated in Figure 5-9.

Figure 5-9 shows that I'_{DHAN} is very strongly correlated with the Input I'_s , with correlations coefficients of 0.973 and 0.993 for the ISTD and SRD datasets respectively. Based on these correlations, we conclude that $SSIM(I'_s, I'_f)$ and $SSIM(I'_{\text{DHAN}}, I'_f)$ have the same cause for lower SSIM in non-shadow regions: a difference between the Input I'_s and GT I'_f .



(a) DHAN structural similarity versus its components on the ISTD dataset



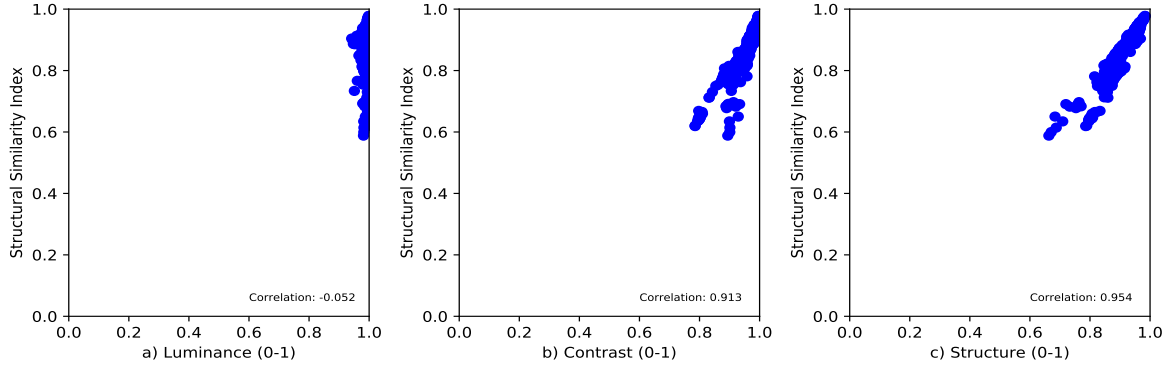
(b) DHAN structural similarity versus its components on the SRD dataset

Figure 5-7: $SSIM(I'_{\text{DHAN}}, I'_f)$ for the (a) ISTD and (b) SRD datasets. The SSIM metric is split into three components; (a) Luminance, (b) Contrast, and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.

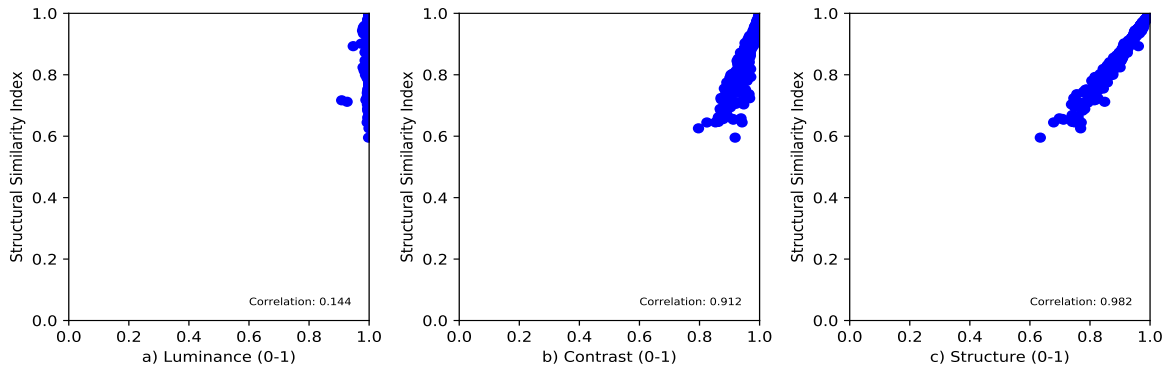
Second, we evaluate the SGAN. Figures 5-8a and 5-8b show that the overall structural similarity $SSIM(I'_{\text{SGAN}}, I'_f)$ is lower than the input. Furthermore, we see that the SGAN structural similarity luminance element (ISTD: -0.052, SRD: 0.144) has a negligible correlation with the SGAN SSIM, but both the contrast (ISTD: 0.913, SRD: 0.912) and structure (ISTD: 0.954, 0.982) elements have strong correlations with the SSIM. Since the contrast correlations increase to very strong correlations compared to the Input, the SGAN must change the contrast of I'_s .

To confirm that the Input and GT dissimilarities do not cause the contrast changes in the SGAN and thereby our hypothesis from Subsection 5-1-2, we evaluate the correlation between the SGAN and Input in Figure 5-9. The SGAN correlation with the Input is moderate for the ISTD (0.530) and SRD (0.674) datasets. The structural dissimilarity of the SGAN in non-shadow regions cannot be explained by the difference between I'_s and I'_f .

To determine if this alteration in non-shadow regions negatively or positively affects the SSIM, we determine the amount of images in the datasets that the SGAN increases the SSIM of in non-shadow regions. We use the following formula to calculate the amount of images for



(a) SGAN structural similarity versus its components on the ISTD dataset



(b) SGAN structural similarity versus its components on the SRD dataset

Figure 5-8: $SSIM(I'_{SGAN}, I'_s)$ for the (a) ISTD and (b) SRD datasets. The SSIM metric is split into three components; (a) Luminance, (b) Contrast, and (c) Structure. The contribution of each of the components is plotted against the total SSIM of each image.

which the SGAN decreases the SSIM:

$$\sum_1^N S^{(i)} \begin{cases} S^{(i)} = 1 & \text{if } \frac{SSIM(I'_s^{(i)}, I'_f^{(i)})}{SSIM(I'_{SGAN}, I'_f)} > 1 \\ S^{(i)} = 0 & \text{if } \frac{SSIM(I'_s^{(i)}, I'_f^{(i)})}{SSIM(I'_{SGAN}, I'_f)} \leq 1 \end{cases}$$

For 93% (ISTD) and 91% (SRD) of the datasets, the SGAN decreases the structural similarity outside the shadow regions. Concluding, we state that the SGAN changes pixel values in non-shadow regions, decreasing the contrast component of the SSIM and, therefore, the overall SSIM in non-shadow regions.

Masksize

Figure 5-10 shows the masksize of the images is plotted against its structural similarity for all removal methods to evaluate the correlation of the masksize with the structural similarity. The masksize's correlation with the input's structural similarity is negligible for the ISTD (0.024) and SRD (0.088) datasets. The correlation between the DHAN and masksize is weak

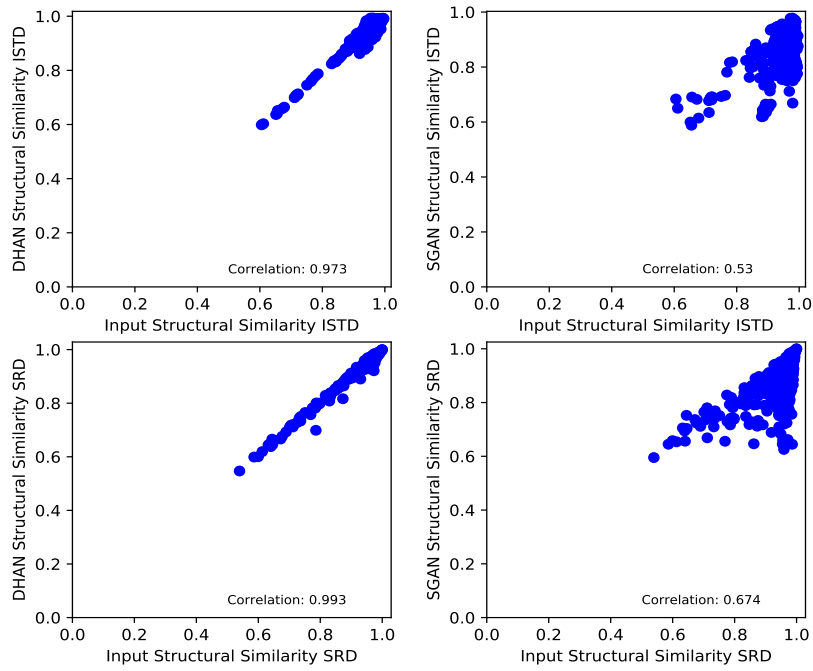


Figure 5-9: Input $SSIM(I'_s, I'_f)$ plotted against $SSIM(I'_{DHAN}, I'_f)$ and $SSIM(I'_{SGAN}, I'_f)$ for the ISTD (top row) and SRD (bottom row) datasets.

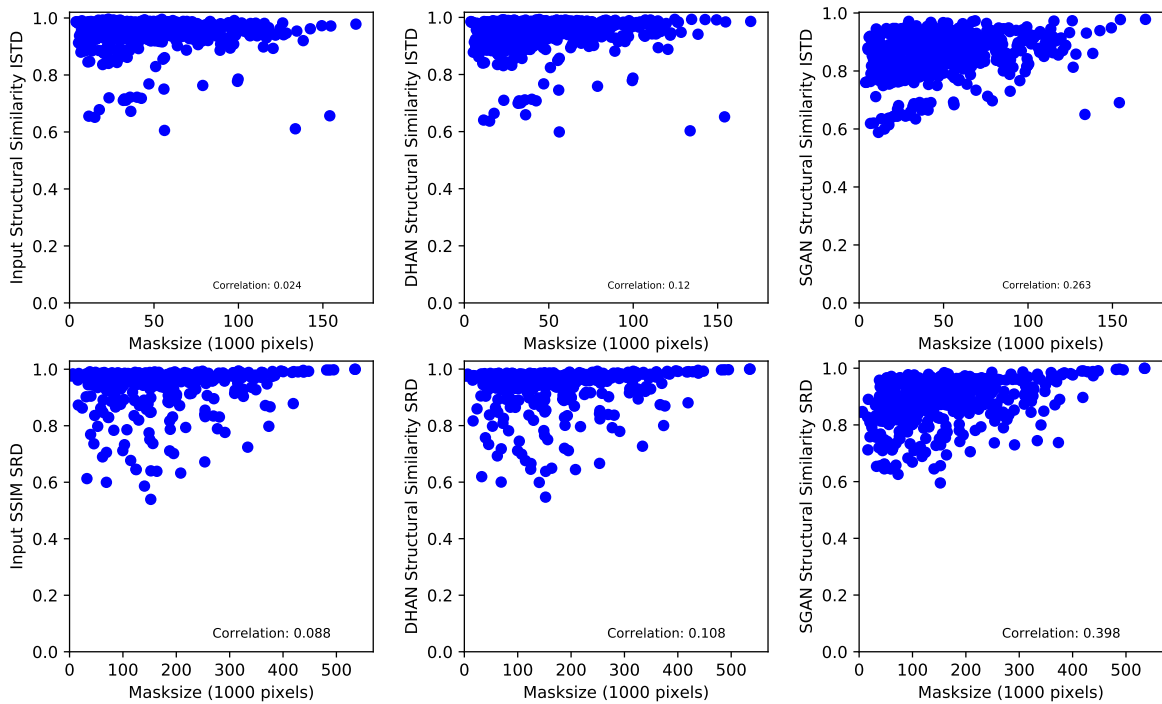


Figure 5-10: Correlation between structural similarity and masksize

for the ISTD (0.120) and SRD (0.108) datasets and weak between the SGAN and masksize

for the ISTD (0.263) and SRD (0.398) datasets. The masksize does not significantly affect the structural similarity outside of the shadow regions, and we conclude that the size of the mask does not cause the decreased structural similarity in non-shadow regions.

5-2 Segmentation performance

In this section, the DHAN and SGAN, trained on the ISTD and SRD datasets, are used for shadow removal on the CamVid segmentation dataset I_c . The performance metrics used are the PA and mIoU as explained in Chapter 4.

5-2-1 Improvement ratio analysis

To evaluate the improvement of segmentation after shadow removal, we compare the segmentation PA before and after shadow removal. To make an improvement comparison per image, we calculate the improvement ratio for the PA. We define the PA improvement ratio as:

$$PA_{\text{ratio}} = \frac{PA_{\text{net}}}{PA_{\text{segnet}}} \quad (5-2)$$

Where PA_{net} is the PA of the DHAN or SGAN shadow removed images, trained on the ISTD or SRD dataset after segmentation by SegNet and PA_{segnet} is the PA of SegNet without any shadow removal before segmentation.

In Figure 5-11a, the distribution of the PA improvement ratio is shown. For the DHAN network trained on the ISTD and SRD dataset, the medians and quartiles are located below an improvement ratio of 1. The PA improves for 15% and 12% of the images, respectively. Since the PA is decreased for the majority of the images, we conclude that the DHAN trained on the ISTD and SRD datasets does not benefit the segmentation performance of SegNet with shadow removal based on the PA metric.

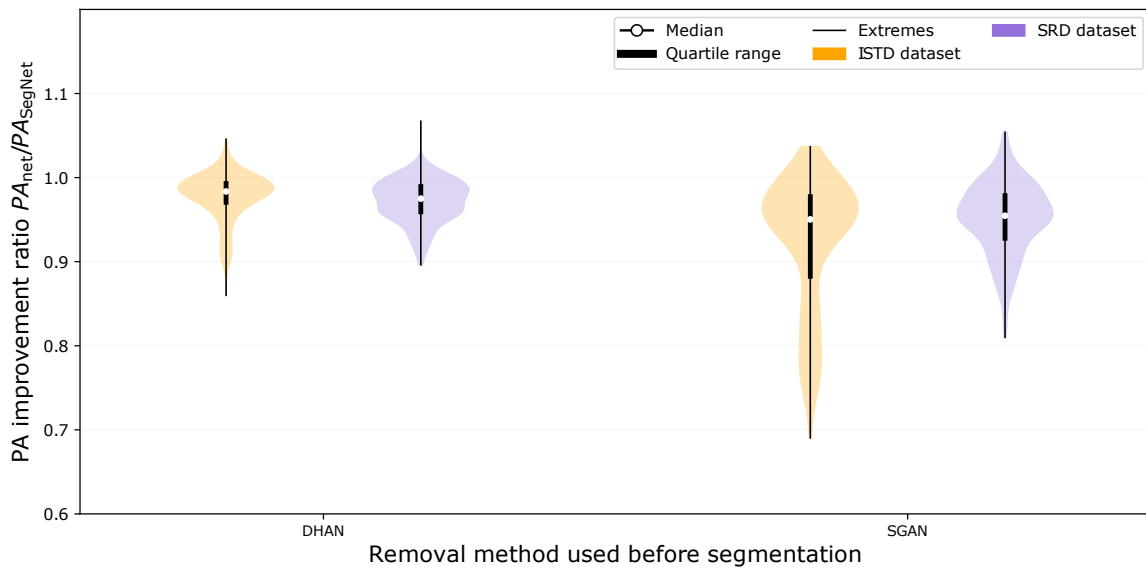
For the SGAN, the same is observed as for the DHAN; the medians and quartiles are located below an improvement ratio of 1, with an improvement ratio over one for 10% and 8% of the images for the ISTD and SRD-trained networks, respectively. The SGAN trained on the ISTD and SRD dataset decreased the PA performance of SegNet for even more images than the DHAN, rendering these trained SGAN models unfit to increase the segmentation performance based on the PA.

The second performance metric used for evaluation is the mIoU. Again, the improvement ratio is defined as:

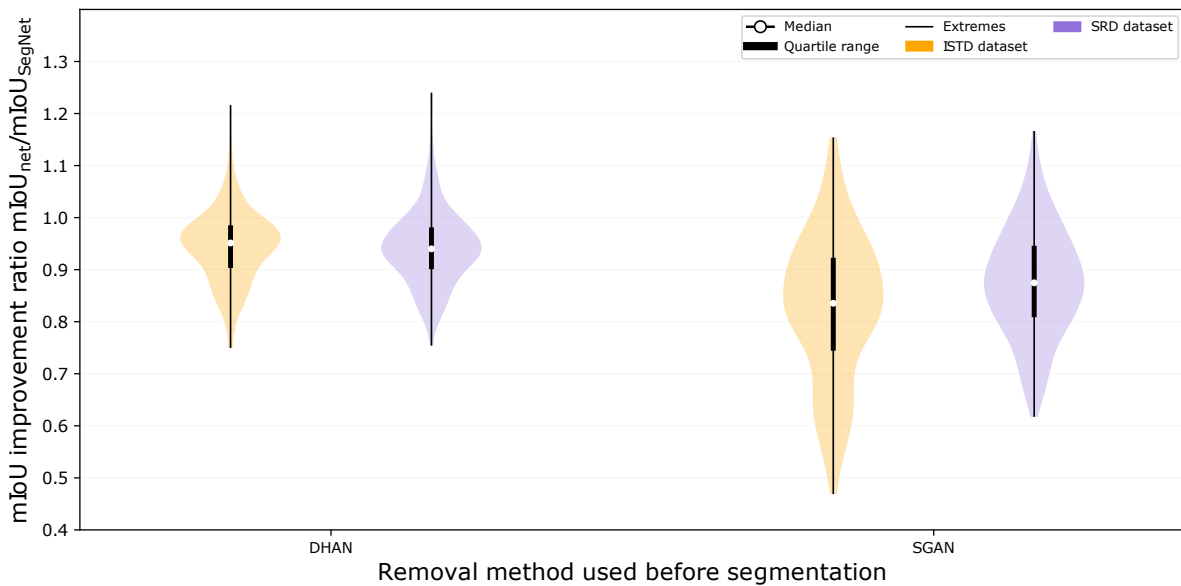
$$mIoU_{\text{ratio}} = \frac{mIoU_{\text{net}}}{mIoU_{\text{segnet}}} \quad (5-3)$$

Where $mIoU_{\text{net}}$ is the mIoU of the DHAN or SGAN shadow removed images, trained on the ISTD or SRD dataset after segmentation by SegNet and $mIoU_{\text{segnet}}$ is the mIoU of SegNet without any shadow removal on the images.

In Figure 5-11b, we see the distribution of the improvement ratio of the mIoU. For the



(a) Improvement ratio of the Pixel Accuracy metric



(b) Improvement ratio of the Intersection over Union metric

Figure 5-11: Improvement ratios of the (a) PA and (b) mIoU on the CamVid dataset. The improvement ratios are shown for the DHAN and SGAN network trained on the ISTD and SRD datasets.

DHAN and SGAN, all quartiles and medians are below an improvement ratio of 1. The DHAN trained on the ISTD dataset improves the mIoU for 20% of the images and trained on the SRD dataset for 19% of the images, while the SGAN improves the mIoU for 10% and 8% of the CamVid dataset. Both the DHAN and SGAN decrease the segmentation accuracy for the majority of the images.

5-2-2 Visual inspection

To analyze why shadow removal using the DHAN or SGAN does not benefit segmentation based on the PA and mIoU metrics, we visually inspect the CamVid images before and after shadow removal by the DHAN and SGAN. This visualization is shown in Figures 5-12a and 5-12b. The intensity differences have been calculated according to Section 4-3-2

Based on visual inspection of these figures, the observation is made that the DHAN and SGAN increase the pixel values and, with that, the brightness over large parts of the images. Based on the difference maps, we observe that both methods mainly increase the pixel values of the darker areas of the images, such as the road at the bottom of each image. The increase of pixel intensity over multiple parts of the image might influence the performance of SegNet and, instead of increasing the segmentation performance, decrease it. To check this hypothesis, we evaluate the correlation between the images' mean intensity difference and the segmentation improvement. These correlations are shown in Figure 5-13.

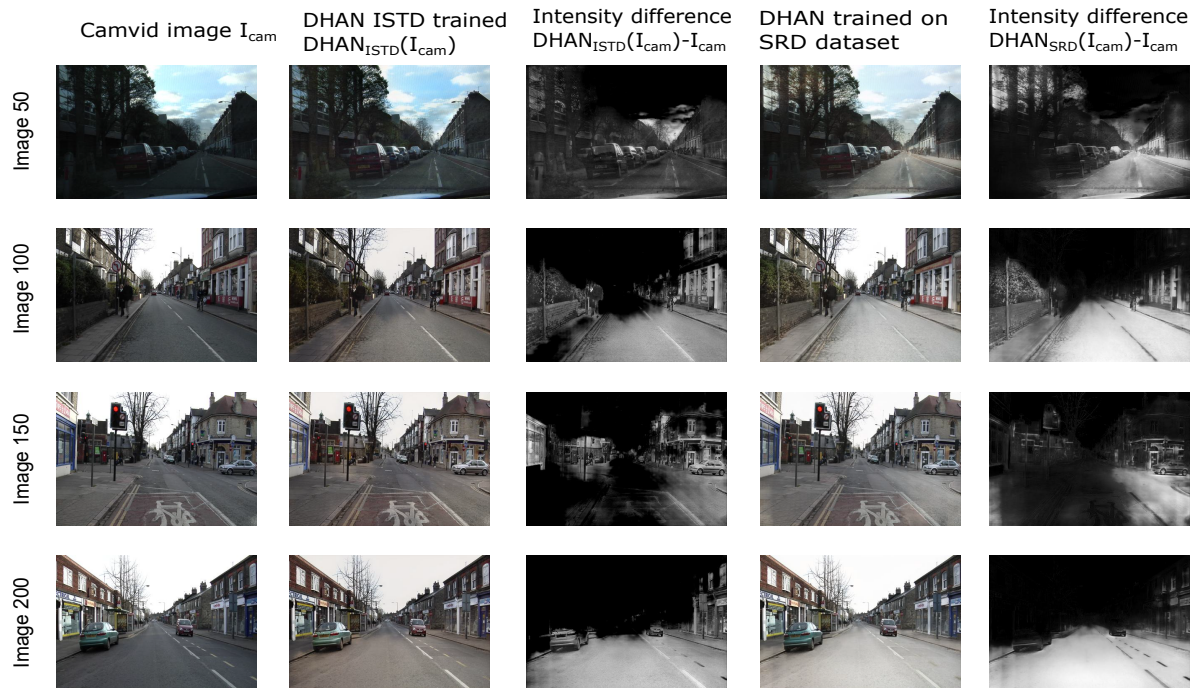
The DHAN and SGAN trained on the ISTD dataset show a strong negative correlation (-0.689 and -0.82, respectively) between the intensity increase and improvement ratio. Increasing the intensity of an image the PA compared to SegNet PA without shadow removal. The DHAN and SGAN trained on the SRD dataset also have strong negative correlations (-0.673 and -0.724, respectively) between the intensity increase and PA.

The DHAN and SGAN, trained on either dataset, show a negative correlation between this intensity increase and performance ratio, concluding that the DHAN and SGAN cannot increase segmentation performance over the CamVid dataset by increasing the intensity.

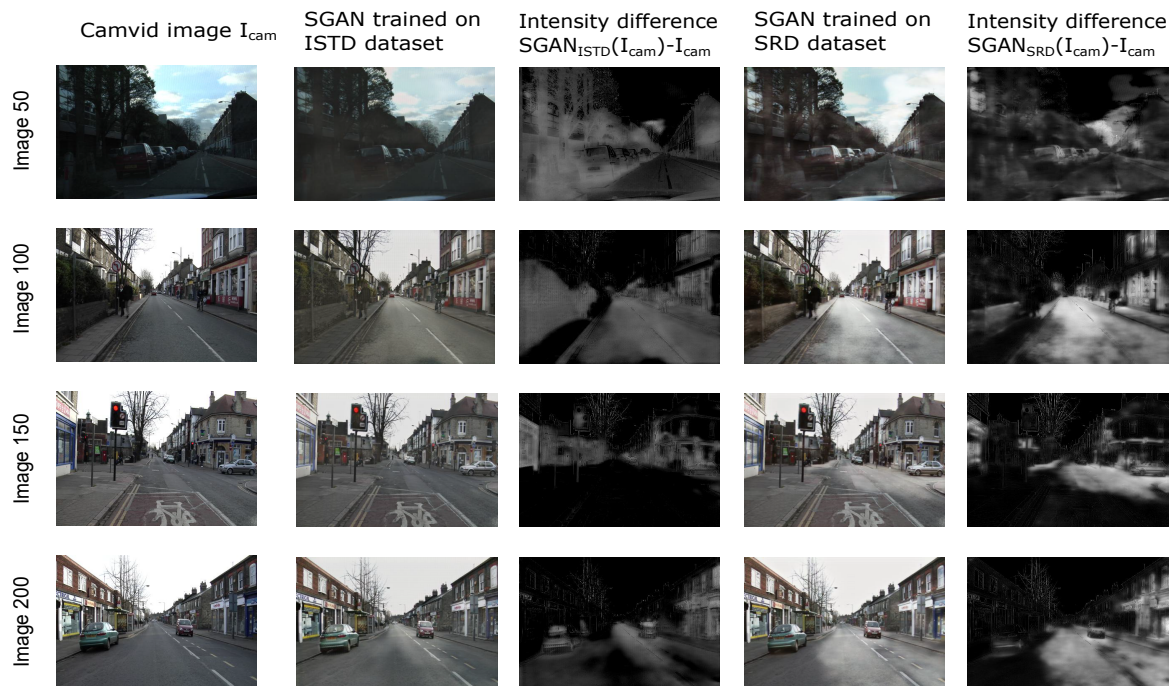
5-3 Shadow analysis

In this section, we create synthetic shadow datasets based on the shadows in the ISTD and SRD datasets. In the previous section, we have seen that the DHAN and SGAN increase the pixel intensity of the images in the CamVid dataset and that this increase negatively correlates with the PA and mIoU improvement ratios. A reason that might cause the intensity increase instead of only shadow removal is that the large shadows and relatively simple background in the ISTD and SRD are too dissimilar from the CamVid images. The DHAN and SGAN networks cannot grasp the more complex data in the CamVid images correctly based on their ISTD and SRD training. In Figures 5-4 and 5-5, it can be seen that the ISTD and SRD dataset images have simple backgrounds compared to the complex city environments in the CamVid dataset. To learn the DHAN and SGAN to remove shadows in environments depicted in the CamVid dataset, a dataset of CamVid images with synthetically added shadows is created for training and testing.

The creation of this dataset starts with an analysis of what shadow is. This analysis is based on the ISTD and SRD datasets. The pixel difference between the shadow-free images I_f and shadow images I_s is calculated according to Section 4-3-2. In Figure 5-14, we see a visual representation of the pixel difference.



(a) DHAN shadow removal on CamVid dataset



(b) SGAN shadow removal on CamVid dataset

Figure 5-12: The 50'th, 100'th, 150'th and 200'th of the CamVid dataset images I_{cam} . The post shadow removal (a) DHAN and (b) images are depicted, including the difference map between the shadow removed and the original CamVid image. The intensity difference maps are shown in grayscale.

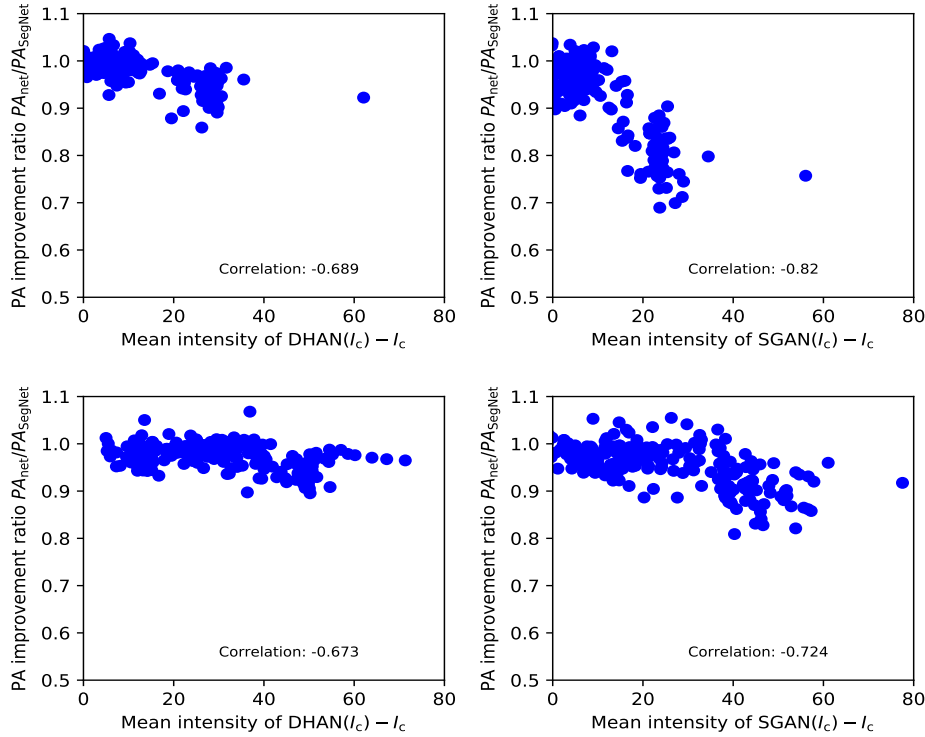


Figure 5-13: Mean intensity of $NET(I_c) - I_c$ plotted against the PA performance ratio for the DHAN and SGAN networks trained on both the ISTD and SRD dataset.

The intensity difference states how much the pixel values in shadow regions differ from pixels in the ground truth, where no shadow is seen. The intensity difference is used to define the shadow intensity. The shadow intensity is defined as how much darker the shadow is compared to the same region in the shadow-free image. To create a synthetic dataset, the intensity difference found in the ISTD and SRD datasets is subtracted from selected regions in the CamVid dataset, creating shadows in those specified regions. Since the intensity difference is not the same for each image and varies, the distribution of the mean intensity difference is determined for each color channel for both the ISTD and SRD datasets. This distribution is shown in Figure 5-15.

Ten datasets with increasing shadow intensity are created to evaluate the effect of different intensities on shadow removal and segmentation performance. The intuition behind this method comes from the fact that not each shadow has the same intensity and that the intensity of each shadow can differ. In Section 4-3-2, the method for creating the synthetic datasets is stated. In this same section, a formula for generating the synthetic datasets is shown. The increase in shadow intensity in this formula is based on the distribution shown in Figure 5-15. The first synthetic dataset $DataSyn_1$ is created by subtracting the tenth quantile intensity average of both datasets (red: 50, green: 44, blue: 39) from the CamVid images I_c in the shadow regions. The shadow regions are defined by the set of masks M_c . For each dataset $DataSyn_k$, we increase the shadow intensity by $[6.2, 5.9, 5.6]$ for the red, green, and blue channels, respectively. The final dataset $DataSyn_{10}$ will have a shadow intensity of $[112, 103, 95]$ (red, green, and blue channel, respectively), equal to the intensity at the ninetieth quantile average of the distributions in Figure 5-15. This is mathematically stated in Section 4-3-2,

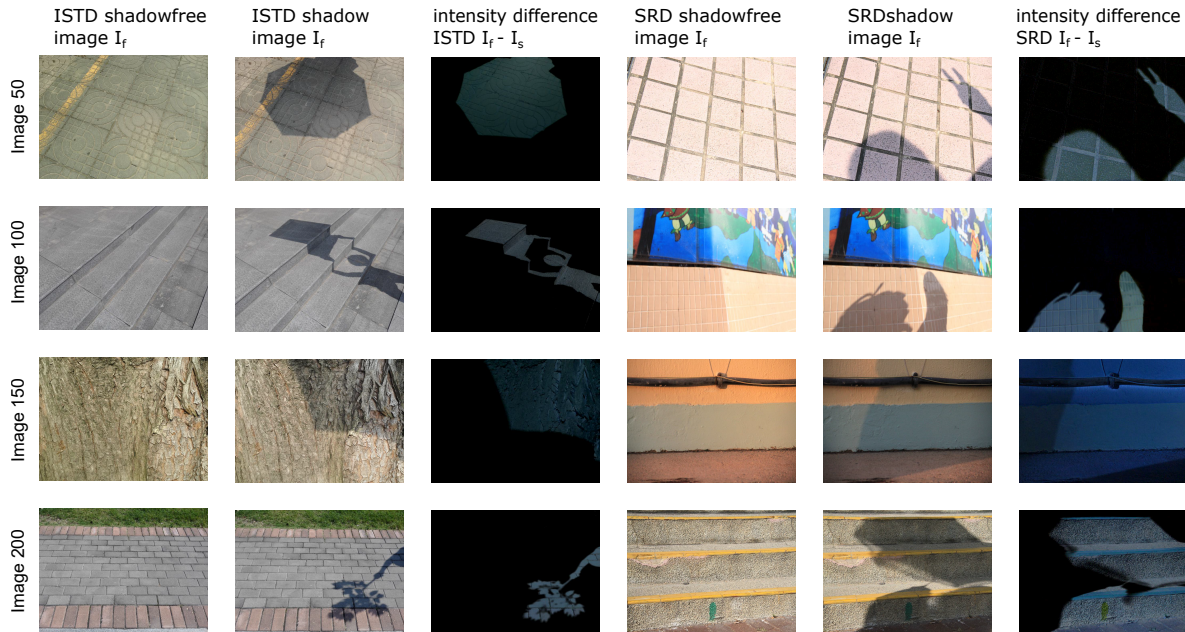


Figure 5-14: The 50'th, 100'th, 150'th and 200'th of ISTD and SRD datasets respectively. The shadow image is subtracted from the shadow-free image to find the intensity difference.

Equation 4-15. In Appendix B-2, an example of the CamVid images with increasing shadow intensity is shown.

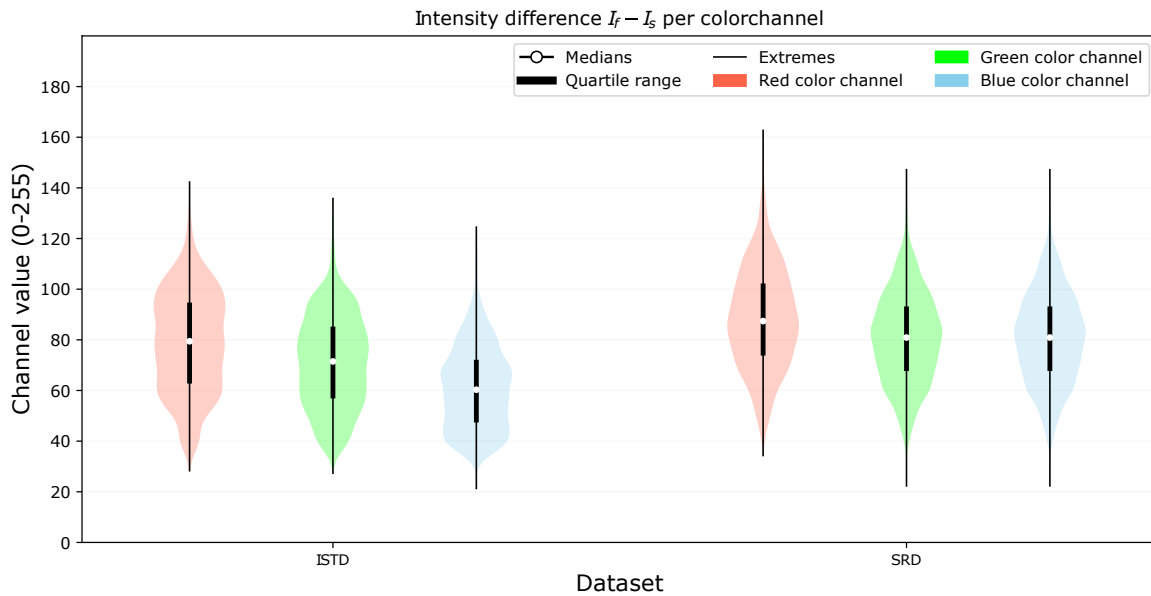


Figure 5-15: Pixel intensity difference I_{diff} distribution for the ISTD and SRD dataset. The pixel intensity difference is discrete and can only be an integer value.

5-4 Results on synthetic shadows

The DHAN is retrained on the synthetic datasets DataSyn_i , generating eleven networks, according to Section 4-3-2. The eleven networks DHAN_i are evaluated on shadow removal using the SSIM improvement ratio as metric, similarly to Section 5-2. The networks are evaluated on the complete set of shadow intensity ranges in DataSyn_{11} . Then, the segmentation performance improvement ratio is calculated for each of the trained networks, similarly to Section 5-2.

5-4-1 Structural similarity comparison

In Figure 5-16, the structural similarity improvement ratio is shown for all DHAN_i models on the DataSyn_{11} dataset. The DHAN_6 network has the highest number of images with a SSIM improvement ratio larger than one (94%) and has the highest mean (1.030) improvement ratio. The DHAN_i trained on the darkest shadow (DataSyn_{10}) increase the structural similarity for the smallest percentage of images (83%). Furthermore, we observe that the number of images with an improvement ratio over 1 between DHAN_6 and DHAN_{10} gradually decreases.

The DHAN_6 network also has the median with the largest value (1.027). The DHAN_{10} and DHAN_1 networks have the lowest median (1.022 and 1.021, respectively). An observation is that for the median and mean increase for value between DHAN_1 and DHAN_6 , and decreases between DHAN_6 and DHAN_{10} . The DHAN_2 network is the exception. This network has a higher median and mean than DHAN_3 . The model trained on all shadow intensities has a median of 1.022 and mean of 1.025. It does not outperform the DHAN_6 network. In Appendix B-1, a complimentary table is given containing the median, mean, quartile and images improved percentage.

A reason for the DHAN_6 network performing best in terms of median and total images improved could be due to the intensity of the training set of each network and the test set. The hypothesis is that the datasets trained on the lighter shadows are less able to remove the dark shadows, and networks trained on darker shadows are less able to remove the lighter shadows. To verify this hypothesis, the SSIM improvement ratio is calculated for the dataset with the lightest (DataSyn_1) and darkest shadows (DataSyn_{10}). The results are depicted in Figure 5-17.

For the DataSyn_1 dataset, shown in Figure 5-17 by the red violins, the percentage of images with an improvement ratio larger than one is 86% for both the DHAN_1 and DHAN_2 networks. This percentage decreases for each following network unto DHAN_{10} , which has an improvement ratio larger than one for 54% of the images. The median of each network its performance ratio distribution decreases from DHAN_2 (1.013) to DHAN_5 (1.007). DHAN_6 and DHAN_7 have the same median as DHAN_5 . The DHAN_{10} network has the lowest median, at 1.002.

The median, mean and percent of images improved all show a very strong negative correlation with the shadow intensity (-0.959, -0.904, -0.960, respectively). This correlation shows that if we increase the shadow intensity used for training the networks, the median, mean

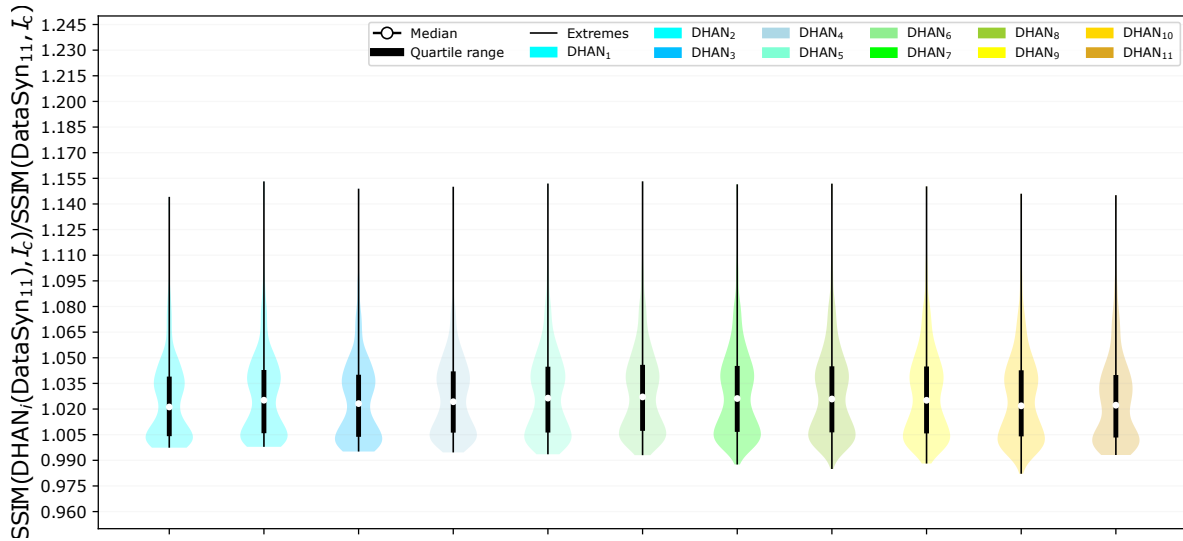


Figure 5-16: Structural similarity improvement ratio of each of the trained DHAN_i networks on the DataSyn_{11} dataset.

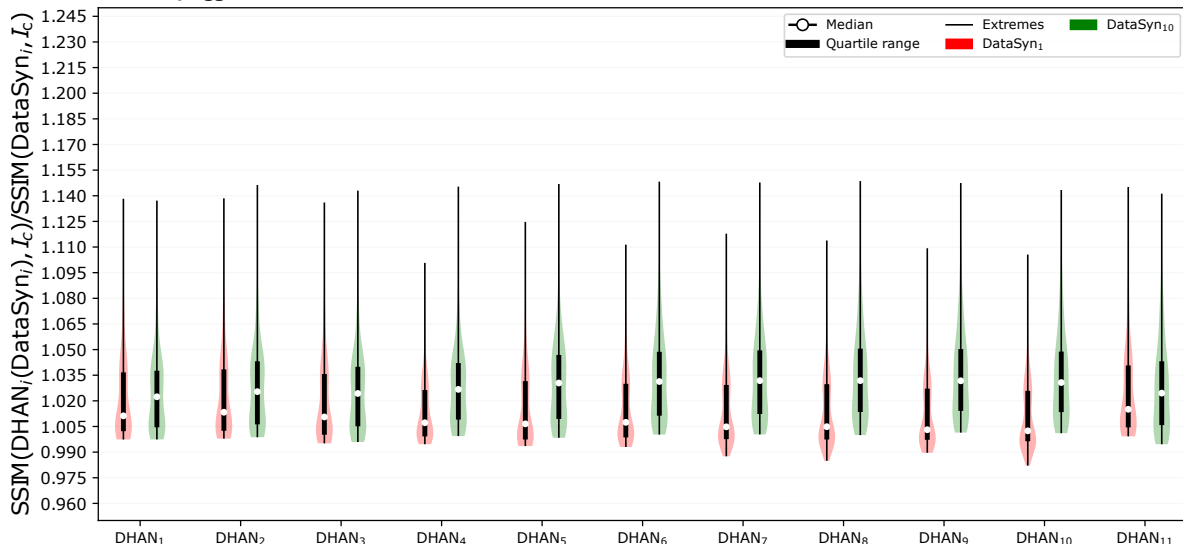


Figure 5-17: Structural similarity improvement ratio of each of the trained DHAN_i networks on the DataSyn_1 and DataSyn_{10} datasets.

and the number of images with an improvement ratio larger than one decrease when tested on DataSyn_1 . This leads to the conclusion that if the darkness of the shadow in the dataset used for training the network increases, the percentage of images with improved structural similarity, mean, and the median of the structural similarity decreases if tested on DataSyn_1 .

The same analysis is performed for the DataSyn_{10} dataset, containing the images with the darkest shadows, shown in Figure 5-17 by the green violins. The percentage of images with an improvement ratio larger than one is 88% for DHAN_1 , and 93% for DHAN_2 . DHAN_3 shows a drop to 86%. DHAN_4 and DHAN_5 (both 95%) are the last networks that do not have an improvement ratio above one for 100% of the images. Evaluating the medians, we see

that the median increases between DHAN_1 (1.022) and DHAN_9 (1.032), with an exception of DHAN_3 (1.024).

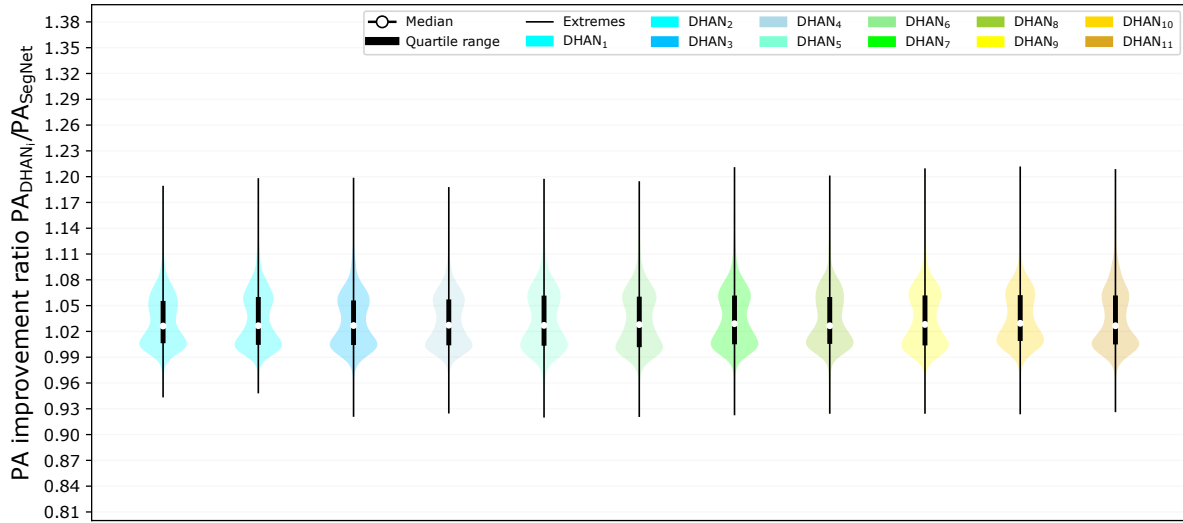
The median, mean and percent of images improved all show a strong to very strong correlation with the shadow intensity (0.933, 0.955 and 0.853, respectively). This correlation shows that if we increase the shadow intensity used for training the networks, the median, mean and the number of images with an improvement ratio larger than one increase when tested on DataSyn_{10} . This leads to the conclusion that if the darkness of the shadow in the dataset used for training the network increases, the percentage of images with improved structural similarity, mean, and the median of the structural similarity increase if tested on DataSyn_{10} .

The evaluation of the structural similarity improvement ratio evaluated on DataSyn_{11} has shown that the DHAN_6 improves the structural similarity of most images, and its distribution has the largest median. Furthermore, it is observed that the medians of the distribution and percentage of images with improved structural similarity decrease if the shadows in the training set become lighter or darker compared to DHAN_6 . The shadow intensity of the training set is hypothesized to be the cause. This hypothesis is confirmed by evaluating the performance of each network DHAN_i on the datasets with the lightest and darkest shadow and finding the correlation between the shadow intensity and the medians, means and number of images with increased structural similarity. The result shows that the hypothesis is correct: the percentage of images with improved structural similarity declines if the intensity difference between the shadows in the test dataset (DataSyn_1 or DataSyn_{10}) and training set DataSyn_i increases. This is confirmed by the correlations. In Appendix B-1, complementary tables are given that summarize the medians, means and percentage of images with an improvement ratio larger than one for all DHAN_i networks.

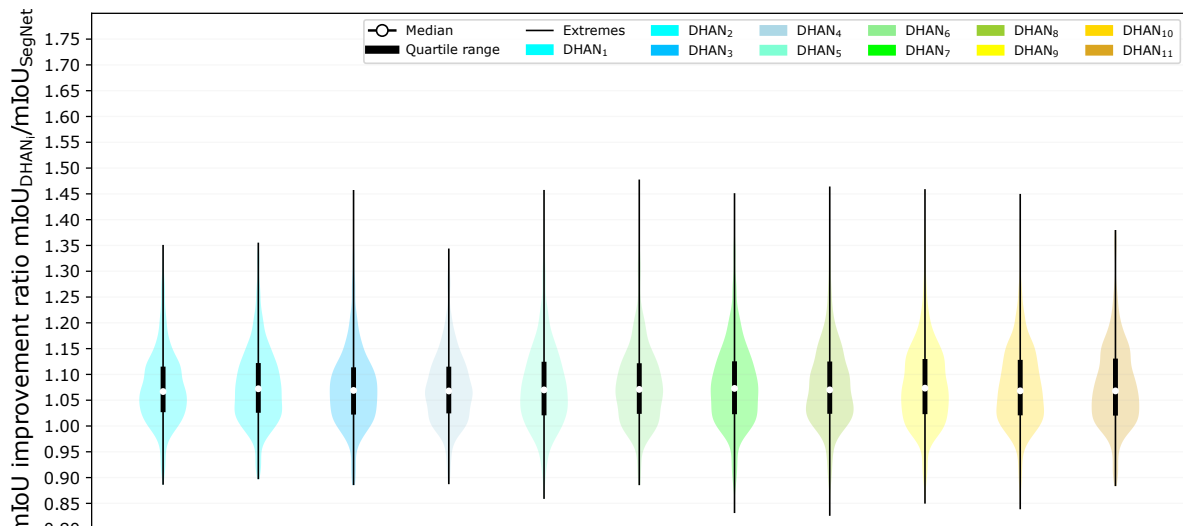
5-4-2 Segmentation performance

To assess the accuracy increase of the segmentation after shadow removal, we evaluate the PA and mIoU metrics compared to the accuracy without shadow removal. To this extent, we use the improvement ratio for the PA and mIoU as in Section 5-2. In Figure 5-18a, the improvement ratio of the pixel accuracy is shown. The DHAN_{10} network shows a PA improvement ratio over one for most images (91%), meaning that it increases the PA for most images compared to no shadow removal. This network also has the largest median (1.030). The DHAN_3 network increases the PA for the lowest number of images (83%) and has the lowest median (1.026). The network with the best structural similarity improvement from the previous section, the DHAN_6 network, improves the PA for 85% of the images, with its median at 1.028. The correlation between the shadow intensity and the median and mean (0.870 and 0.800, respectively), shows us that increasing the shadow intensity of the training set has a positive effect on the segmentation accuracy evaluated with the PA.

In Figure 5-18b, the improvement ratio for the mIoU is shown. The DHAN_8 network improves the mIoU for most images (88%). The DHAN_9 network has the highest valued median (1.073), and the best mIoU improvement is achieved by the DHAN_6 network (1.48). The least number of images with an mIoU improvement ratio over one are generated by the The



(a) Improvement ratio of the Pixel Accuracy metric



(b) Improvement ratio of the mean Intersection over Union metric

Figure 5-18: Improvement ratios of the (a) PA and (b) mIoU. The ratios have been calculated according to Equations 5-2 and 5-3. The improvement ratio is shown for each retrained $DHAN_i$ network.

$DHAN_4$ network (84%). The $DHAN_1$ network has the lowest median (1.066). The correlation between the shadow intensity used for training and the median is 0.87, and the correlation for the mean is 0.628. The correlation is not as strong as for the PA metric, but still shows a positive connection between the mIoU improvement ratio and shadow intensity used for training.

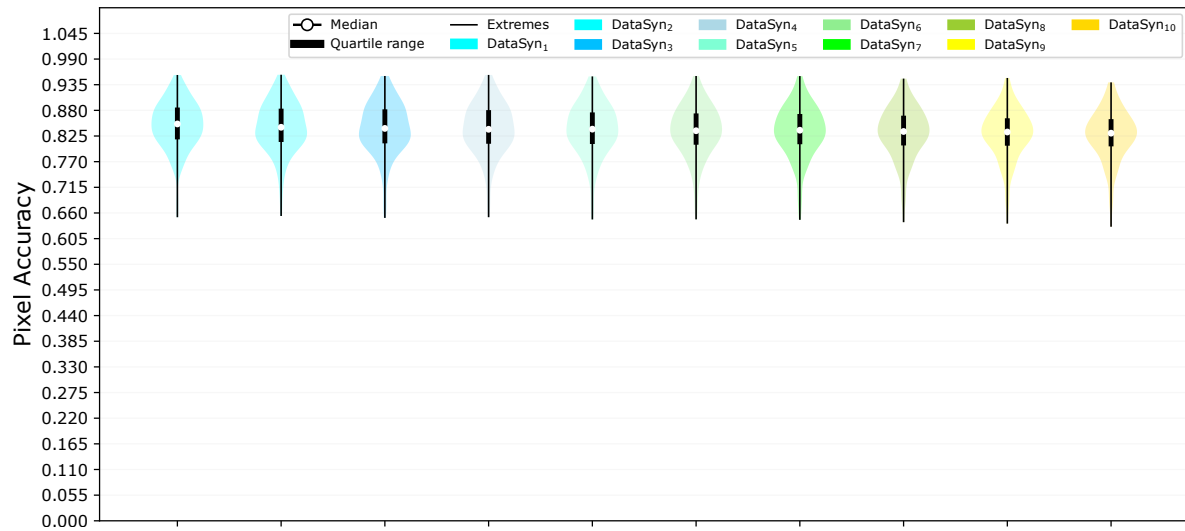
For both PA and mIoU, the most images improved, and largest median and mean values are achieved by the networks trained on the darker shadows. The shadow intensity and network

performance have a strong positive correlation for the PA and moderate to strong correlation for the mIoU. A possible explanation could be because the segmentation is affected more negatively by darker shadows. Therefore, the networks trained to remove darker shadows are able to improve the performance more. To validate this hypothesis, the PA and mIoU of the non-shadow removed images are evaluated. Suppose the images with darker shadows have a lower PA and mIoU. In that case, it can be argued that shadow removal on lower-intensity shadows does not benefit the PA and mIoU as much as the removal of higher-intensity shadows, since there is a strong positive correlation between the shadow intensity used for training and ability to remove the darkest shadows, as we have seen in Section 5-4-1. In Appendix B-1, complementary tables are given that summarize the medians, means, minima, maxima, and percentage of images with an improvement ratio larger than one for all DHAN_i networks for both the PA and mIoU metric.

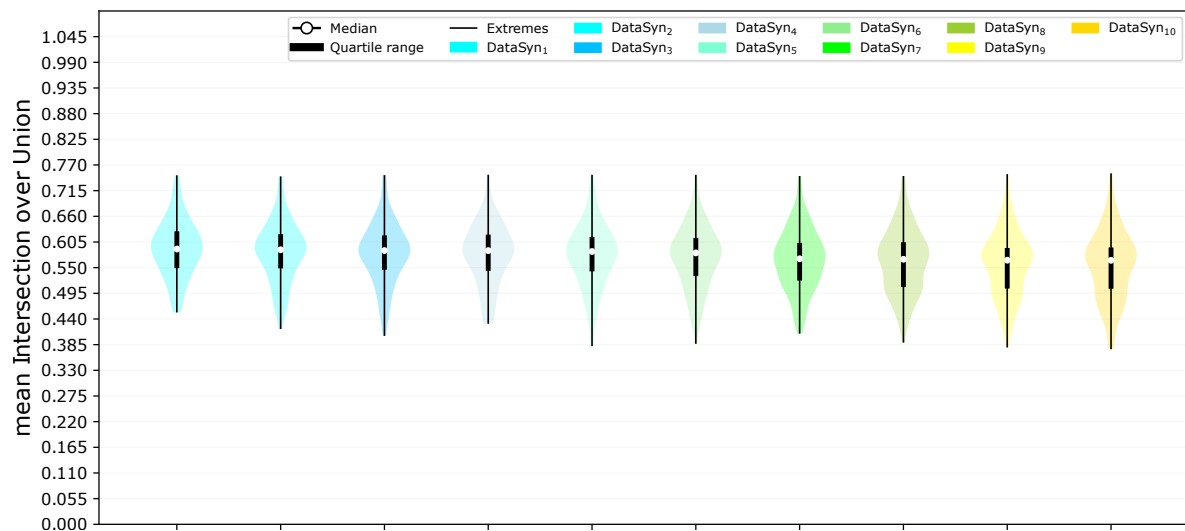
In Figure 5-19a, the PA distribution is shown for the DataSyn_i datasets with $i = 1, 2, \dots, 10$. We observe from this figure that the median (0.851), maximum (0.956) and minimum (0.651) of DataSyn_1 have the highest values. The median values decrease for each following DataSyn_i until reaching their lowest value (0.831) for the DataSyn_{10} dataset, moving the body of the distributions toward lower PA scores for increasing i . The shadow intensity of DataSyn_i and the PA show strong to very strong negative correlation for the median (-0.934), minima (-0.927), maxima (-0.853) and mean (-0.991). This correlation shows that increasing shadow intensity negatively affects the PA.

In Figure 5-19b, the mIoU distribution is shown for the DataSyn_i datasets with $i = 1, 2, \dots, 10$. Using the same approach as for the PA evaluation, we observe that the median (0.590), maximum (0.748) and minimum (0.454) of DataSyn_1 have the highest values. The median values decrease for each following DataSyn_i until reaching their lowest value (0.566) for the DataSyn_{10} dataset. This is reflected by the fact that the body of the distributions is moving toward lower mIoU values for increasing i . The strong negative correlation between the shadow intensity of DataSyn_i and the median (-0.940), and mean (-0.993) shows that SegNet is not able to segment darker shadows as well as lighter shadows. We conclude that an increasing shadow intensity negatively affects the mIoU. In Appendix B-1, complementary tables are given that summarize the medians, means, minima and maxima of the PA and mIoU accuracy for the DataSyn_i datasets.

Combining the results from Figures 5-18a to 5-19b, we can conclude that all DHAN_i networks increase the PA for a minimum of 83% of the images and the mIoU for a minimum of 84% of the images. The DHAN_{10} improves the PA for most images, while the DHAN_8 network improves the mIoU for most images. The hypothesis that segmentation accuracy based on the PA and mIoU is affected more negatively by darker shadows was proven, explaining why the DHAN_i networks trained on darker shadows increase the accuracy according to these metrics for more images. In Section 5-4-1, it is shown that higher-intensity shadows are best removed by a network trained on a dataset containing higher-intensity (darker) shadows and vice versa. These results show that the networks trained on a larger shadow intensity are able to increase the PA of more images and why networks trained on shadows with higher intensity are able to increase the segmentation accuracy for the largest number of images in DataSyn_{11} .



(a) Pixel Accuracy without shadow removal



(b) mean Intersection over Union without shadow removal

Figure 5-19: Pixel Accuracy and Intersection over Union segmentation accuracy on the DataSyn_{*i*} datasets before shadow removal.

Chapter 6

Conclusion

This chapter finalizes this thesis with a conclusion of the main results. Section 6-1 gives a summary of the results in Chapter 5. In Section 6-2, an answer to the research question and sub-questions is given. Then, the limitations to this work are discussed. Based on these results and limitations, Section 6-3 proposes future work to improve on the findings of this thesis.

6-1 Summary of results

During this thesis, shadow removal to improve the segmentation accuracy has been researched. The thesis is motivated by the decrease in segmentation performance in adverse lighting conditions, that can hard the traversability estimation in autonomous vehicles. Changes in lighting conditions when driving through a changing environment cast shadows that can negatively affect the segmentation of the image depicting the environment surrounding an autonomous vehicle. This decreases the ability of the autonomous vehicle to generate a correct prediction of the traversability of the terrain.

Applying a shadow removal method to images before segmentation could improve the ability of the Unmanned Ground Vehicle (UGV) to segment its surroundings correctly. To this extent, two shadow removal methods have been compared: the supervised Dual Hierarchical Aggregation Network (DHAN) and unsupervised Shadow Generative Adversarial Network (SGAN). The networks have been trained on the Image Shadow Triplets Dataset (ISTD) and Shadow Removal Dataset (SRD) datasets to compare their shadow removal performance. The metric to evaluate their performance is the Structural Similarity Index Measure (SSIM).

The results of the SSIM evaluation of both networks show that the DHAN network improves the structural similarity of 99.6% of the images in the ISTD dataset. For the SRD dataset, the DHAN improves the structural similarity of 100% of the images. The SGAN does not achieve the same improvement percentage as the DHAN. It improves the structural similarity

of 63.3% and 85.7% of the images for the ISTD and SRD datasets, respectively. The median structural similarity increases by 0.12, from 0.82 to 0.94, for the DHAN removal method. The SGAN shows a small 0.01 improvement, from 0.82 to 0.83. Furthermore, further analysis of the images in non-shadow regions has shown that the SGAN changes pixel values that are not in the shadow region. Therefore, the supervised DHAN is selected as the preferred shadow removal method.

In a second experiment, the DHAN and SGAN shadow removal approaches are applied to the CamVid dataset. After shadow removal, the images are segmented using Segmentation Network (SegNet). The results are evaluated using the Pixel Accuracy (PA) and mean Intersection over Union (mIoU) metrics. Shadow removal with the DHAN or SGAN before segmentation increases the segmentation performance for most images when trained on the ISTD dataset. The PA increases for 15% of the images and the Intersection over Union (IoU) for 20% of the images. The SGAN improves the PA and IoU for 10% of the images. It is concluded that for the rest of the images, the performance decreases, and shadow removal with the DHAN or SGAN negatively influences the segmentation accuracy.

It is hypothesized that the ISTD and SRD shadows do not resemble the CamVid dataset, creating DHAN and SGAN networks that are not able to generalize well enough to remove shadow present in the CamVid dataset. To improve the ability to remove shadows in environments depicted in the CamVid dataset, a synthetic shadow dataset is generated based on the shadows in the ISTD and SRD datasets. The DHAN is retrained on this synthetic dataset, and the segmentation accuracy is re-evaluated, again using the PA and IoU metrics.

The final experiment results evaluate the shadow removal, using the SSIM metric, and semantic segmentation accuracy. The structural similarity analysis shows that the networks trained on the synthetic data, $DHAN_i$, increase the structural similarity for at least 83% of the dataset. The $DHAN_6$ network, trained on the $DataSyn_6$ dataset, is able to increase the structural similarity for 94% of the images. The segmentation PA and IoU, increases for at least 83% and 84% of the images, respectively. By evaluating the correlations between the segmentation accuracy metrics and shadow intensity, we find that shadow intensity and segmentation accuracy have a strong negative correlation, meaning that darker shadows decrease the segmentation accuracy more than lighter shadows. Therefore, the networks trained on darker shadows increase the segmentation accuracy more than lighter shadows.

6-2 Conclusion

Segmentation accuracy after shadow removal has been researched in this thesis. The main research question is formulated as follows:

"How can shadow removal improve semantic segmentation accuracy for traversability estimation?"

To answer the research question, first the sub-questions are answered:

"1. How can shadow be removed without changing pixel values in non-shadow regions of an image?"

Generative Adversarial Networks are a popular tool for image translation, that can be used to translate a shadow image into a shadow-free image. The generator takes a shadow image as input and translates this to a non-shadow image, where the discriminator tries to distinguish between the generated shadow-free image and the Ground Truth shadow-free image.

The supervised DHAN and unsupervised SGAN are networks for shadow removal. In the first set of experiments, both networks are trained on the ISTD and SRD datasets. After training, the structural similarity is calculated before shadow removal, and after shadow removal using the DHAN and SGAN.

To show if the DHAN and SGAN change pixel values outside of the shadow region, the pixel values of the masks are set to zero outside of the shadow regions. Setting the masks to zero gives us the ability to evaluate the changes in pixel values outside the shadow regions. We observe that the structural similarity of the Input and DHAN is very strongly correlated, with a minimal correlation of 0.973. This shows that the DHAN does not change pixel values outside the shadow region. For the SGAN, we observe that the Input and SGAN structural similarities show moderate to strong correlation, with a maximum correlation of 0.674. This shows that the SGAN does changes pixel values, and therefore the structural similarity outside the shadow regions.

"2. How can the effect of shadow removal on semantic segmentation be compared to semantic segmentation without shadow removal?"

Semantic segmentation is a popular method for pixel-wise image classification. In autonomous vehicles, this pixel-wise classified image can be used for traversability estimation. The evaluation of the effect of shadow removal on the segmentation accuracy is conducted in two stages.

In the first stage, the effect of shadow removal is evaluated using the structural similarity improvement ratio. The structural similarity improvement ratio shows the relationship between the structural similarity before and after shadow removal. Using the improvement ratio it is shown that the DHAN is able to increase the structural similarity for 99.6% of the images when trained on the ISTD dataset, and 100% of the images when trained on the SRD dataset. The SGAN is able to increase the structural similarity for 62.3% and 85.7% of the images, for the ISTD and SRD dataset, respectively.

In the second stage, the increase in segmentation accuracy is calculated using the PA and mIoU improvement ratios. These ratios show the relationship between the PA and mIoU of the segmentation before and after shadow removal. Using this ratio it is shown that the PA improves for a maximum of 15% of the images when the DHAN network trained on the ISTD dataset is used for shadow removal. The mIoU improves for a maximum of 20% of the images when the DHAN network trained on the ISTD dataset is used for shadow removal.

"How can shadow removal improve semantic segmentation accuracy for traversability estimation?"

From the answer of sub-question 2, it can be seen that shadow removal using the DHAN or SGAN decreases semantic segmentation accuracy for most images. The DHAN and SGAN networks are not able to transfer their shadow removal abilities from the ISTD or SRD dataset onto the CamVid dataset. The difference in background between the ISTD or SRD and Cambridge-driving Labeled Video Database (CamVid) dataset might be the reason. To this extent, new datasets are generated based on the shadows in the ISTD and SRD datasets. Based on the results of sub-question one, the DHAN is used for further shadow removal on these created datasets.

Trained on the new datasets, the DHAN is evaluated on the PA and mIoU improvement ratios. The best DHAN networks increase the PA and mIoU for 91% and 88% of the images. This proves that shadow removal increases segmentation accuracy for a majority of the images when trained on shadows in images resembling the CamVid dataset.

Furthermore, it is shown that the intensity of the shadow affects the segmentation performance. Increasing the shadow intensity decreases the PA and mIoU accuracy. The DHAN networks trained on darker shadows are able to increase the PA and mIoU for more images.

Concluding, semantic segmentation accuracy of SegNet improves after shadow removal performed by the DHAN network trained on a dataset with synthetic shadows. This also shows the drawback of this approach. The DataSyn_i datasets are generated by approximating shadows based on the ISTD and SRD datasets. This creates synthetic shadows that try to mimic real world shadows. They are, however, still artificially generated and cannot replace an evaluation with real world data. To improve on this and other aspects of this research, several recommendations for future work are proposed.

6-3 Future work

Real world data

The final experiment uses a synthetic dataset to evaluate the segmentation performance after shadow removal. Using the synthetic dataset, the concept of shadow removal for increased segmentation accuracy is shown. This is, however, just an approximation of reality. To test the method for real-world usage, a dataset containing real-world data of shadow and shadow-free image pairs captured in complex environments as that of the CamVid dataset is required. This dataset can then be used to perform the same analysis in this thesis. This way, the conclusion that DHAN shadow removal before semantic segmentation using the SegNet can be tested.

Virtual data

A solution to the absence of a shadow removal dataset with complex environments that an autonomous vehicle would encounter, could be a virtually generated dataset. This dataset can be used to train the shadow removal network and the segmentation performance increase or decrease can be evaluated. This is an alternative approach to the synthetically generated

dataset, and might prove able to better understand the shadows in real-world images. This dataset can also be used to validate the results of this thesis.

Segmentation accuracy as loss

The shadow removal networks are trained for the sole purpose of shadow removal. The fact that the images generated by these networks are used for segmentation, is not incorporated in the shadow removal networks their structure or losses. By adding the PA or mIoU as loss to the loss functions of the shadow removal networks, the networks might be able to generate images better suited for semantic segmentation, improving the segmentation accuracy.

Additional information

The shadow removal methods in this thesis are trained to remove shadows using Red, Green, Blue (RGB) images. By adjusting the architecture of the shadow removal networks, extra information can be incorporated. Sensors placed in autonomous vehicles are not limited to RGB cameras. Extra information obtained from these sensors, such as LiDAR or radar, could improve the ability of the shadow removal networks when generating shadow free images. The improved shadow removal could lead to a further increase in segmentation accuracy.

Multiple semantic segmentation methods

In this research, the SegNet network for segmentation is used to determine the segmentation accuracy. This limits the results to SegNet. Evaluating the segmentation accuracy of multiple segmentation networks could gain more insight into whether shadow removal for increased segmentation accuracy only applies to SegNet or multiple segmentation networks.

Appendix A

Additional figures

A-1 Activation function supplement

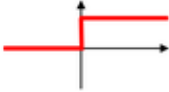
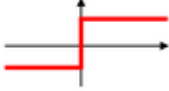
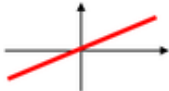


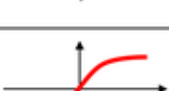
Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

Figure A-1: More complete overview of activation functions and their graphs. This figure states the activation function, its equation, its use in Neural Networks and its visualisation in 1D graph form. From: [19].

A-2 ADAM learning rate scheduler

The ADAM learning rate scheduler algorithm can be pseudo coded as [78]. :

ALGORITHM ADAM:

Require: α : Stepsize
 Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
 Require: $f(\theta)$: Stochastic objective function with parameters θ
 Require: θ_0 : Initial parameter vector
 $m_0 \leftarrow 0$ (Initialize 1st moment vector)
 $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
 $t \leftarrow 0$ (Initialize timestep)
 while θ_t not converged do
 $t \leftarrow t + 1$
 $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
 $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
 $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
 $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
 $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
 $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)
 end while
 return θ_t (resulting parameters)

Appendix B

Complimentary tables and figures

B-1 Complimentary tables

Table B-1: Complimentary table to Figure 5-16. The structural similarity improvement ratio median, quartile, mean, and percentage of images improved values for the DHAN_i networks on the DataSyn_{11} dataset. The correlation between DHAN_1 to DHAN_{10} and the shadow intensity is shown in the eleventh column.

	DHAN_1	DHAN_2	DHAN_3	DHAN_4	DHAN_5	DHAN_6	DHAN_7	DHAN_8	DHAN_9	DHAN_{10}	Correlation	DHAN_{11}
Median	1.021	1.025	1.023	1.024	1.026	1.027	1.026	1.026	1.025	1.022	0.238	1.022
Quartile 1	1.004	1.006	1.003	1.006	1.006	1.007	1.007	1.006	1.006	1.004	0.023	1.003
Quartile 3	1.039	1.043	1.040	1.042	1.045	1.046	1.045	1.045	1.045	1.043	0.651	1.040
Mean	1.025	1.028	1.026	1.027	1.029	1.030	1.030	1.029	1.029	1.027	0.540	1.025
% Increase	88	92	85	92	88	94	92	90	88	83	-0.223	86

Table B-2: Complimentary table to Figure 5-17. The structural similarity improvement ratio median, mean, and percentage of images improved values for the DHAN_i networks on the DataSyn_1 ($i = 1$) and DataSyn_{10} ($i = 10$) datasets. The correlation between DHAN_1 to DHAN_{10} and the shadow intensity is shown in the eleventh column.

	DHAN_1	DHAN_2	DHAN_3	DHAN_4	DHAN_5	DHAN_6	DHAN_7	DHAN_8	DHAN_9	DHAN_{10}	Correlation	DHAN_{11}
Median ($i=1$)	1.011	1.013	1.010	1.008	1.007	1.007	1.007	1.004	1.003	1.002	-0.959	1.015
Mean ($i=1$)	1.022	1.023	1.020	1.014	1.016	1.016	1.015	1.014	1.013	1.011	-0.904	1.025
% Increase ($i=1$)	86	86	76	67	66	67	62	60	55	54	-0.96	85
Median ($i=10$)	1.022	1.025	1.024	1.027	1.030	1.031	1.031	1.032	1.032	1.032	0.933	1.025
Mean ($i=10$)	1.025	1.028	1.026	1.029	1.031	1.033	1.034	1.035	1.036	1.035	0.955	1.028
% Increase ($i=10$)	88	93	86	95	95	100	100	100	100	100	0.853	85

Table B-3: Complimentary table to Figure 5-18a. The PA improvement ratio median, mean, minimum, maximum, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_{11}$ dataset. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.

	DHAN ₁	DHAN ₂	DHAN ₃	DHAN ₄	DHAN ₅	DHAN ₆	DHAN ₇	DHAN ₈	DHAN ₉	DHAN ₁₀	Correlation	DHAN ₁₁
Median	1.026	1.027	1.027	1.027	1.027	1.028	1.029	1.028	1.028	1.030	0.870	1.027
Min	0.943	0.948	0.920	0.925	0.920	0.920	0.920	0.920	0.920	0.920	-0.719	0.930
Max	1.189	1.198	1.199	1.188	1.198	1.195	1.211	1.201	1.210	1.212	0.783	1.209
Mean	1.033	1.034	1.032	1.033	1.034	1.033	1.035	1.035	1.036	1.037	0.800	1.035
% Increase	87	86	83	86	85	86	85	89	89	91	0.661	86

Table B-4: Complimentary table to Figure 5-18b. The mIoU improvement ratio median, mean, minimum, maximum, and percentage of images improved values for the $DHAN_i$ networks on the $DataSyn_{11}$ dataset. The correlation between $DHAN_1$ to $DHAN_{10}$ and the shadow intensity is shown in the eleventh column.

	DHAN ₁	DHAN ₂	DHAN ₃	DHAN ₄	DHAN ₅	DHAN ₆	DHAN ₇	DHAN ₈	DHAN ₉	DHAN ₁₀	Correlation	DHAN ₁₁
Median	1.066	1.072	1.068	1.068	1.070	1.071	1.073	1.073	1.073	1.071	0.874	1.068
Min	0.886	0.897	0.885	0.887	0.859	0.886	0.831	0.836	0.850	0.839	-0.828	0.883
Max	1.351	1.355	1.458	1.344	1.458	1.478	1.451	1.464	1.459	1.350	0.343	1.378
Mean	1.074	1.079	1.074	1.074	1.077	1.078	1.080	1.080	1.080	1.081	0.628	1.078
% Increase	87	87	87	84	86	86	86	88	87	87	0.187	87

Table B-5: Complimentary table to Figure 5-19a. The median, mean, minimum, and maximum PA for the $DataSyn_i$ datasets without shadow removal. The correlation between $DataSyn_1$ to $DataSyn_{10}$ PA values and the shadow intensity is shown in the eleventh column.

	DataSyn ₁	DataSyn ₂	DataSyn ₃	DataSyn ₄	DataSyn ₅	DataSyn ₆	DataSyn ₇	DataSyn ₈	DataSyn ₉	DataSyn ₁₀	Correlation
Median	0.851	0.844	0.841	0.839	0.839	0.836	0.837	0.835	0.834	0.831	-0.934
Min	0.651	0.654	0.649	0.651	0.646	0.646	0.645	0.640	0.637	0.630	-0.927
Max	0.956	0.956	0.954	0.956	0.952	0.953	0.954	0.948	0.949	0.940	-0.853
Mean	0.849	0.844	0.842	0.840	0.839	0.837	0.835	0.832	0.830	0.827	-0.991

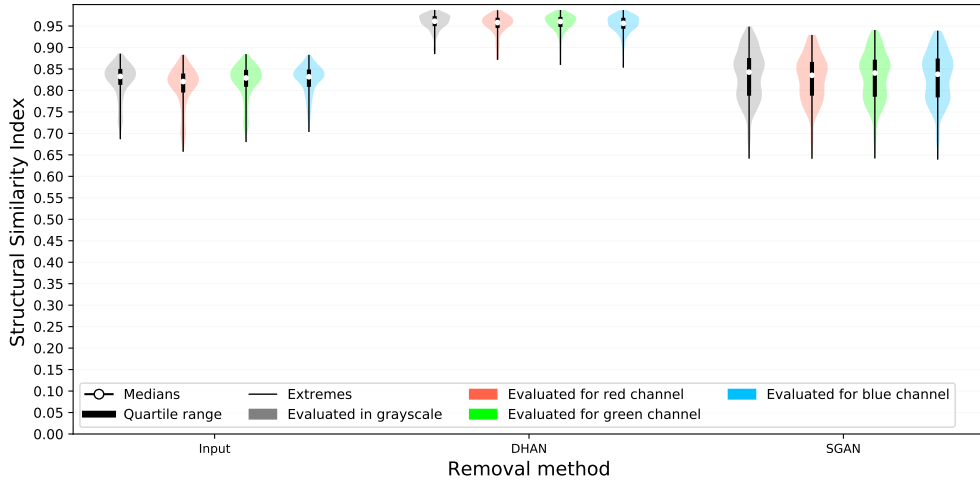
Table B-6: Complimentary table to Figure 5-19b. The median, mean, minimum, and maximum mIoU for the $DataSyn_i$ datasets without shadow removal. The correlation between $DataSyn_1$ to $DataSyn_{10}$ mIoU values and the shadow intensity is shown in the eleventh column.

	DataSyn ₁	DataSyn ₂	DataSyn ₃	DataSyn ₄	DataSyn ₅	DataSyn ₆	DataSyn ₇	DataSyn ₈	DataSyn ₉	DataSyn ₁₀	Correlation
Median	0.590	0.588	0.587	0.587	0.585	0.582	0.569	0.568	0.566	0.566	-0.940
Min	0.454	0.418	0.404	0.428	0.382	0.387	0.408	0.389	0.379	0.375	-0.81
Max	0.784	0.746	0.748	0.749	0.749	0.749	0.746	0.746	0.750	0.752	-0.447
Mean	0.591	0.584	0.580	0.578	0.575	0.572	0.567	0.564	0.557	0.554	-0.993

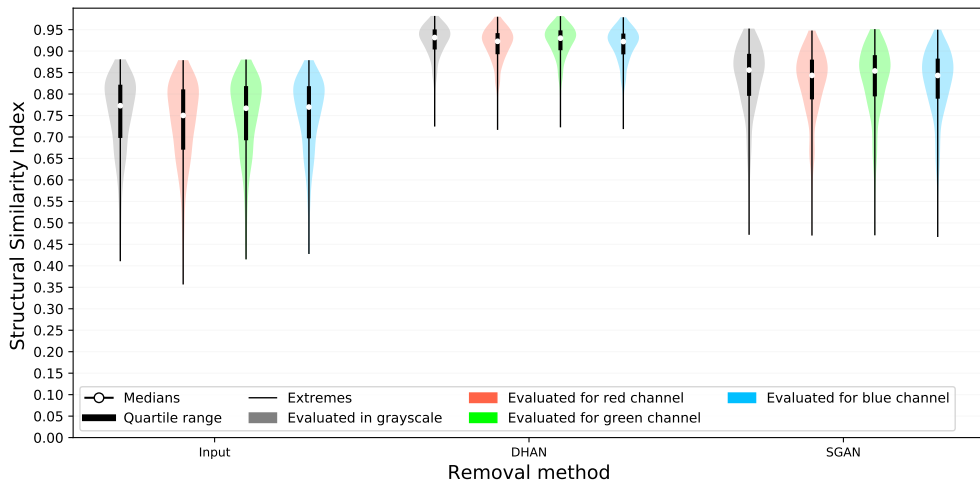
B-2 Complimentary figures



Figure B-1: Visualisation of increasing shadow intensities on CamVid image, where the shadow intensity increases from from left to right, top to bottom.



(a) Structural similarity between the Ground Truth and Input-, DHAN- and SGAN shadow removed images for the ISTD dataset



(b) Structural similarity between the Ground Truth and Input-, DHAN- and SGAN shadow removed images for the SRD dataset

Figure B-2: Structural similarity between the Ground Truth and Input-, DHAN - and SGAN shadow removed images for the (a) ISTD and (b) SRD datasets where images $SSIM(I'_s, I'_f) < 0.90$ are removed. For all three removal methods, the structural similarity is shown in grayscale (gray violins) and for each of the three color channels (red, green, and blue violins).

Bibliography

- [1] Waymo, “Waymo 1.” <https://waymo.com/waymo-one/>.
- [2] A. Amini, “Introduction to deep learning,” March 2022. <http://introtodeeplearning.com/>.
- [3] N. Sharma, V. Jain, and A. Mishra, “An analysis of convolutional neural networks for image classification,” *Procedia computer science*, vol. 132, pp. 377–384, 2018.
- [4] I. S. Mohamed, *Detection and Tracking of Pallets using a Laser Rangefinder and Machine Learning Techniques*. PhD thesis, 09 2017.
- [5] M. Yani *et al.*, “Application of transfer learning using convolutional neural network method for early detection of terry’s nail,” in *Journal of Physics: Conference Series*, vol. 1201, p. 012052, IOP Publishing, 2019.
- [6] X. Hou, L. Shen, K. Sun, and G. Qiu, “Deep feature consistent variational autoencoder,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1133–1141, 2017.
- [7] L. Mou and X. Zhu, “Im2height: Height estimation from single monocular imagery via fully residual convolutional-deconvolutional network,” 02 2018.
- [8] R. Imtiaz, T. M. Khan, S. S. Naqvi, M. Arsalan, and S. J. Nawaz, “Screening of glaucoma disease from retinal vessel images using semantic segmentation,” *Computers & Electrical Engineering*, vol. 91, p. 107036, 2021.
- [9] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” *CoRR*, vol. abs/1505.04366, 2015.
- [10] H. Alqahtani, M. Kavakli-Thorne, and G. Kumar, “Applications of generative adversarial networks (gans): An updated review,” *Archives of Computational Methods in Engineering*, vol. 28, pp. 525–552, 2021.

- [11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [12] S. B. Akin, “Bias-variance trade-off and polynomial regression.”
- [13] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” *CoRR*, vol. abs/1511.00561, 2015.
- [14] X. Cun, C. Pun, and C. Shi, “Towards ghost-free shadow removal via dual hierarchical aggregation network and shadow matting GAN,” *CoRR*, vol. abs/1911.08718, 2019.
- [15] J. He, P. Wu, Y. Tong, X. Zhang, M. Lei, and J. Gao, “Bearing fault diagnosis via improved one-dimensional multi-scale dilated cnn,” *Sensors*, vol. 21, no. 21, p. 7319, 2021.
- [16] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in *Computer Vision – ECCV 2014*, pp. 346–361, Springer International Publishing, 2014.
- [17] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *CoRR*, vol. abs/1703.10593, 2017.
- [18] X. Hu, Y. Jiang, C.-W. Fu, and P.-A. Heng, “Mask-ShadowGAN: Learning to remove shadows from unpaired data,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, IEEE, oct 2019.
- [19] “An introduction to artificial neurons.” <http://lovefordatascience.blogspot.com/2017/10/an-introduction-to-artificial-neural.html>.
- [20] T. J. Crayton and B. M. Meier, “Autonomous vehicles: Developing a public health research agenda to frame the future of transportation policy,” *Journal of Transport & Health*, vol. 6, pp. 245–252, 2017.
- [21] M. Massar, I. Reza, S. M. Rahman, S. M. H. Abdullah, A. Jamal, and F. S. Al-Ismael, “Impacts of autonomous vehicles on greenhouse gas emissions—positive or negative?,” *International Journal of Environmental Research and Public Health*, vol. 18, no. 11, 2021.
- [22] F. H. Norris, B. A. Matthews, and J. K. Riad, “Characterological, situational, and behavioral risk factors for motor vehicle accidents: a prospective examination,” *Accident Analysis & Prevention*, vol. 32, no. 4, pp. 505–515, 2000.
- [23] Waymo, “Taking our next step in the City by the Bay,” 3 2022.
- [24] A. Mohamed, J. Ren, M. El-Gindy, H. Lang, and A. Ouda, “Literature survey for autonomous vehicles: sensor fusion, computer vision, system identification and fault tolerance,” *International Journal of Automation and Control*, vol. 12, no. 4, pp. 555–581, 2018.
- [25] Z. Chen and X. Huang, “End-to-end learning for lane keeping of self-driving cars,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1856–1860, IEEE, 2017.

-
- [26] R. Wang, Z. Wang, Z. Xu, C. Wang, Q. Li, Y. Zhang, and H. Li, “A real-time object detector for autonomous vehicles based on yolov4,” *Computational Intelligence and Neuroscience*, vol. 2021, 2021.
- [27] E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional networks for semantic segmentation,” *CoRR*, vol. abs/1605.06211, 2016.
- [28] S. J. Fusic, G. Kanagaraj, K. Hariharan, and S. Karthikeyan, “Optimal path planning of autonomous navigation in outdoor environment via heuristic technique,” *Transportation research interdisciplinary perspectives*, vol. 12, p. 100473, 2021.
- [29] S. H. Young, “Robotic autonomy in complex environments with resiliency (racer),” 2022.
- [30] A. Valada, G. L. Oliveira, T. Brox, and W. Burgard, “Deep multispectral semantic scene understanding of forested environments using multimodal fusion,” in *2016 international symposium on experimental robotics*, pp. 465–477, Springer, 2017.
- [31] J. Giesbrecht, “Global path planning for unmanned ground vehicles,” tech. rep., DEFENCE RESEARCH AND DEVELOPMENT SUFFIELD (ALBERTA), 2004.
- [32] M. T. McCann, K. H. Jin, and M. Unser, “Convolutional neural networks for inverse problems in imaging: A review,” *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 85–95, 2017.
- [33] J. V. Hurtado and A. Valada, “Semantic scene segmentation for robotics,” in *Deep Learning for Robot Perception and Cognition*, pp. 279–311, Elsevier, 2022.
- [34] C. Gsaxner, P. M. Roth, J. Wallner, and J. Egger, “Exploit fully automatic low-level segmented pet data for training high-level deep learning algorithms for the corresponding ct data,” *PLOS ONE*, vol. 14, p. e0212550, 03 2019.
- [35] M. Gruosso, N. Capece, and U. Erra, “Human segmentation in surveillance video with deep learning,” *Multimedia Tools and Applications*, vol. 80, no. 1, pp. 1175–1199, 2021.
- [36] F. Wang and Y. Zhang, “A de-raining semantic segmentation network for real-time foreground segmentation,” *Journal of Real-Time Image Processing*, vol. 18, no. 3, pp. 873–887, 2021.
- [37] A. Pfeuffer and K. Dietmayer, “Robust semantic segmentation in adverse weather conditions by means of fast video-sequence segmentation,” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2020.
- [38] A. Ecins, C. Fermüller, and Y. Aloimonos, “Shadow free segmentation in still images using local density measure,” in *2014 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–8, 2014.
- [39] E. Ackerman, “Korean Competition Shows Weather Still a Challenge for Autonomous Cars,” 8 2022.
- [40] D. Specht, “Probabilistic neural networks and the polynomial adaline as complementary techniques for classification,” *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 111–121, 1990.

- [41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [42] M. A. Nielsen, *Neural networks and deep learning*, vol. 25. Determination press San Francisco, CA, USA, 2015.
- [43] B. Ding, H. Qian, and J. Zhou, “Activation functions and their characteristics in deep neural networks,” in *2018 Chinese Control And Decision Conference (CCDC)*, pp. 1836–1841, 2018.
- [44] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (G. Gordon, D. Dunson, and M. Dudík, eds.), vol. 15 of *Proceedings of Machine Learning Research*, (Fort Lauderdale, FL, USA), pp. 315–323, PMLR, 11–13 Apr 2011.
- [45] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.
- [46] M. Verhaegen and V. Verdult, *Filtering and system identification: a least squares approach*. Cambridge university press, 2007.
- [47] C. M. Bishop, *Pattern Recognition and Machine learning*. Springer, 2006.
- [48] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.04747, 2016.
- [49] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.04747, 2016.
- [50] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.
- [51] S. Zhu and K. Yu, “Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5675–5679, 2017.
- [52] W. Fan, “Deep encoder-decoder neural network for fingerprint image denoising and inpainting,” *CoRR*, vol. abs/2005.01115, 2020.
- [53] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *CoRR*, vol. abs/1804.02767, 2018.
- [54] C. Zuo, J. Qian, S. Feng, W. Yin, Y. Li, P. Fan, J. Han, K. Qian, and Q. Chen, “Deep learning in optical metrology: a review,” *Light: Science & Applications*, vol. 11, no. 1, pp. 1–54, 2022.
- [55] S.-J. Park, H. Son, S. Cho, K.-S. Hong, and S. Lee, “Srfeat: Single image super-resolution with feature discrimination,” pp. 455–471, 09 2018.
- [56] A. Amini, “Generative adversarial networks,” March 2022. <http://introtodeeplearning.com/>.

-
- [57] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [58] V. Badrinarayanan, B. Mishra, and R. Cipolla, “Understanding symmetries in deep networks,” *arXiv preprint arXiv:1511.01029*, 2015.
- [59] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *CoRR*, vol. abs/1502.03167, 2015.
- [60] R. Memisevic, C. Zach, M. Pollefeys, and G. E. Hinton, “Gated softmax classification,” *Advances in neural information processing systems*, vol. 23, 2010.
- [61] S. Taghanaki, K. Abhishek, J. Cohen, J. Cohen-Adad, and G. Hamarneh, “Deep semantic segmentation of natural and medical images: a review,” *Artificial Intelligence Review*, vol. 54, 01 2021.
- [62] G. Brostow, J. Fauqueur, and R. Cipolla, “Semantic object classes in video: A high-definition ground truth database,” *Pattern Recognition Letters*, vol. 30, pp. 88–97, 01 2009.
- [63] Q. Chen, J. Xu, and V. Koltun, “Fast image processing with fully-convolutional networks,” *CoRR*, vol. abs/1709.00643, 2017.
- [64] A. L. Maas, “Rectifier nonlinearities improve neural network acoustic models,” 2013.
- [65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017.
- [66] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” *CoRR*, vol. abs/1603.08155, 2016.
- [67] J. Wang, X. Li, and J. Yang, “Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1788–1797, 2018.
- [68] L. Qu, J. Tian, S. He, Y. Tang, and R. W. H. Lau, “Deshadownet: A multi-context embedding deep network for shadow removal,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2308–2316, 2017.
- [69] Y. Xia, D. He, T. Qin, L. Wang, N. Yu, T. Liu, and W. Ma, “Dual learning for machine translation,” *CoRR*, vol. abs/1611.00179, 2016.
- [70] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [71] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [72] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” *CoRR*, vol. abs/1502.01852, 2015.

-
- [73] D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,” *CoRR*, vol. abs/1411.4734, 2014.
- [74] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [75] W. Zhao, H. Zhang, Y. Yan, Y. Fu, and H. Wang, “A semantic segmentation algorithm using fcn with combination of bslic,” *Applied Sciences*, vol. 8, p. 500, 03 2018.
- [76] S. Seo, *A review and comparison of methods for detecting outliers in univariate data sets*. PhD thesis, University of Pittsburgh, 2006.
- [77] P. Schober, C. Boer, and L. A. Schwarte, “Correlation coefficients: appropriate use and interpretation,” *Anesthesia & Analgesia*, vol. 126, no. 5, pp. 1763–1768, 2018.
- [78] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

Glossary

List of Acronyms

NN	Neural Network
ReLU	Rectified Linear Unit
FNN	Feedforward Neural Network
MLP	Multilayer Perceptron Network
MSE	Mean Squared Error
ML	Machine Learning
CNN	Convolutional Neural Network
GAN	Generative Adversarial Network
RGB	Red, Green, Blue
DL	Deep Learning
IoU	Intersection over Union
mIoU	mean Intersection over Union
DHAN	Dual Hierarchical Aggregation Network
CamVid	Cambridge-driving Labeled Video Database
ISTD	Image Shadow Triplets Dataset
SSIM	Structural Similarity Index Measure
SGAN	Shadow Generative Adversarial Network
SRD	Shadow Removal Dataset
ISTD	Image Shadow Triplets Dataset
GT	Ground Truth
IQR	Inter Quartile Range
PA	Pixel Accuracy
SegNet	Segmentation Network

SPP	Spatial Pooling Pyramid
SGD	Stochastic Gradient Descent
DARPA	Defence Advanced Research Projects Agency
RACER	Robotic Autonomy in Complex Environments with Resiliency
UGV	Unmanned Ground Vehicle