Generative Adversarial Networks for Image-to-Image Translation on Street View and MR Images

Simon Karlsson & Per Welander
Master of Science Thesis in Electrical Engineering & Biomedical Engineering

**Generative Adversarial Networks for Image-to-Image Translation on Street View and MR Images**

Simon Karlsson & Per Welander

LIU-IMT-TFK-A—18/554—SE

**Supervisor:**  
**Martin Danelljan**  
ISY, Linköping University  
**Per Cronvall**  
Veoneer  
**Gustav Jagbrant**  
Veoneer

**Examiner:**  
**Anders Eklund**  
IMT, Linköping University

*Department of Biomedical Engineering  
Linköping University  
SE-581 83 Linköping, Sweden*

Copyright © 2018 Simon Karlsson & Per Welander
Abstract

Generative Adversarial Networks (GANs) is a deep learning method that has been developed for synthesizing data. One application for which it can be used for is image-to-image translations. This could prove to be valuable when training deep neural networks for image classification tasks. Two areas where deep learning methods are used are automotive vision systems and medical imaging. Automotive vision systems are expected to handle a broad range of scenarios which demand training data with a high diversity. The scenarios in the medical field are fewer but the problem is instead that it is difficult, time consuming and expensive to collect training data.

This thesis evaluates different GAN models by comparing synthetic MR images produced by the models against ground truth images. A perceptual study is also performed by an expert in the field. It is shown by the study that the implemented GAN models can synthesize visually realistic MR images. It is also shown that models producing more visually realistic synthetic images not necessarily have better results in quantitative error measurements, when compared to ground truth data. Along with the investigations on medical images, the thesis explores the possibilities of generating synthetic street view images of different resolution, light and weather conditions. Different GAN models have been compared, implemented with our own adjustments, and evaluated. The results show that it is possible to create visually realistic images for different translations and image resolutions.
Acknowledgments

We would like to express our thanks to the Classification team at Veoneer for a great environment to work in, for sharing their experience and inviting us to be a part of their Monday fika. Special thanks goes to Michael Sörsäter, Johan Byttner and Juozas Vaicenavicius for interesting discussions and inputs regarding the work. Our greatest thanks go to our supervisors Per Cronvall and Gustav Jagbrant for their interest and continuous support throughout the work. They accurately answered our questions and took time to explore this exiting field with us. Our interest of making this thesis a combination of the medical and automo-
tive field would not have been possible without Erika Andersson and we would like to give special thanks for her collaboration and always meeting us with a smile at the office.

We also want to thank our academic supervisor Martin Danelljan for the valuable discussions we had throughout the work and for all the feedback on the thesis. Finally, we would like to thank our examiner Anders Eklund for participating as an expert in the perceptual study and for making this project possible.

Linköping, June 2018
Simon Karlsson & Per Welander
# Contents

Notation ix

1 Introduction 1
   1.1 Background ........................................... 1
   1.2 Purpose ............................................. 4
   1.3 Motivation ........................................... 5
   1.4 Delimitations ....................................... 6
   1.5 Contributions ...................................... 6
   1.6 Thesis Outline ..................................... 7

2 Theory and related work 9
   2.1 Convolutional neural networks ..................... 9
   2.2 Loss functions ..................................... 10
   2.3 Optimizers ......................................... 11
   2.4 Residual Blocks .................................. 11
   2.5 Variational autoencoders ......................... 12
   2.6 Generative adversarial networks ................. 12
   2.7 GAN based Image-to-Image Translation using unpaired training data ........................................... 14
   2.8 Data augmentation using GANs .................... 16
   2.9 GANs in medical images ............................. 17
   2.10 High resolution street view images by GANs .... 17

3 Method 19
   3.1 Comparison of unsupervised GANs ................. 19
   3.2 Image-to-Image translation using CycleGAN .... 21
       3.2.1 Training ....................................... 22
       3.2.2 Implementation ................................ 24
   3.3 Image-to-Image translation using UNIT .......... 26
       3.3.1 Model Framework .............................. 28
       3.3.2 Variational autoencoder ..................... 28
       3.3.3 Weight-sharing layers ....................... 29
       3.3.4 GAN components .............................. 29
### Notation

#### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAN</td>
<td>Generative Adversarial Network</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic resonance imaging</td>
</tr>
<tr>
<td>VAE</td>
<td>Variational Autoencoder</td>
</tr>
<tr>
<td>ReLU</td>
<td>Rectified linear unit</td>
</tr>
<tr>
<td>MAE</td>
<td>Mean absolute error</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean squared error</td>
</tr>
<tr>
<td>PSNR</td>
<td>Peak signal noise ratio</td>
</tr>
<tr>
<td>MI</td>
<td>Mutual information</td>
</tr>
</tbody>
</table>

#### Model parameters

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A, B$</td>
<td>Different image domains.</td>
</tr>
<tr>
<td>$a$</td>
<td>Real image from domain $A$.</td>
</tr>
<tr>
<td>$b$</td>
<td>Real image from domain $B$.</td>
</tr>
<tr>
<td>^</td>
<td>Indicates a synthetic image. The letter specifies the image domain, e.g. $\hat{a}$ is a synthetic image in domain $A$.</td>
</tr>
<tr>
<td>Gen or G</td>
<td>Generator network.</td>
</tr>
<tr>
<td>Dis or D</td>
<td>Discriminator network.</td>
</tr>
<tr>
<td>Enc or E</td>
<td>Encoder network.</td>
</tr>
<tr>
<td>E_Z</td>
<td>Encoder network with shared weights between domains.</td>
</tr>
<tr>
<td>DE_Z</td>
<td>Decoder network with shared weights between domains.</td>
</tr>
</tbody>
</table>
This report is the result of a thesis project done at Linköping University in collaboration with Veoneer. The project investigates image-to-image translation using the deep learning method generative adversarial networks. This chapter introduces the studied problem and purpose of the thesis. A background is given followed by research questions and a motivation for the thesis.

1.1 Background

Machine learning refers to algorithms capable of generalizing to new data by learning from available data, instead of acting according to predefined instructions. Another way of explaining it is that the algorithms will perform better on a certain task after experience on training data, than it would without the experience. For example, if you want an algorithm that can recognize handwritten numbers in images, one approach could be to set up rules based on image information. An edge detection algorithm together with certain heuristic rules could be used. The programmer would have to look at some image examples and establish rules that accomplish the intended task. A machine learning approach is instead to feed an algorithm with labeled examples, allowing the algorithm to learn a predictive model. The learned predictive model depends on the training data, and the performance can improve with added training since it is the algorithm itself that constructs the solution to the intended task.

Different machine learning algorithms have been developed for different tasks. Two common examples of tasks are Classification and Regression. In classification tasks the algorithm is expected to specify a label that a certain input belongs to. The input can for example be an image and a label, where the image is a hand-
written digit, and the label is a number between zero and nine. What differs classification models from regression models is the output. A regression model maps specific input data to a continuous variable. These models could for example be used to predict the outside temperature given previous temperature data. Another machine learning task is synthetization of data where the algorithm should generate new data that resembles the training data. Synthetization of data can for example be used in open world video game production. An algorithm capable of generating new worlds based on inspiration from manually created graphical landscapes could alleviate the manual labor, with no limitation on how big the world could get.

Machine learning methods are often divided into *Supervised* and *Unsupervised* methods. This division has to do with the formulation of the learning problem the certain algorithm has. Supervised models are trained using ground truth data corresponding to each input data. This means the algorithm learns with a sort of teacher telling it what is correct. Examples of supervised methods are regression and classification which was mentioned earlier. Unsupervised models do not have this teacher and instead try to find properties and correlations in the training dataset that should also be present in new data. Data synthetization models are commonly unsupervised.

In most machine learning models, training data is of great importance. One particular situation is in deep learning models. Deep learning refers to complex models consisting of multiple layers of linear and non-linear operations, or processing units, where complex representations are learned by expressions of simpler representations in earlier layers. Deep learning methods allow algorithms to learn hierarchies of concepts, where simple concepts combine to more sophisticated concepts, like for example, complex image processing tasks. Generally, a more complex concept requires additional simpler concepts, or stacks of concepts, which is related to additional layers in the deep learning model. Increased model size usually means a larger dataset is needed for training. This makes optimal use of smaller datasets and methods for data augmentation interesting research areas. [10]

A larger dataset will intuitively allow improved performance of any deep learning model. However, the increased dataset size does not guarantee an increased amount of information that the model can benefit from. For example, if adding a number of images that have similar appearance to existing images in the dataset, a performance increase should not be expected. Adding images that increase the diversity should be better. This is because the distribution of scenarios that a developed system is expected to handle must be reflected by the training data. For example, consider a face recognition system. If the system only trains on well illuminated conditions, it will probably perform poorly when the lighting is bad. This problem can be solved with training data of badly illuminated conditions, but if this for some reason is impossible, data synthetization could be a solution.

Means for synthesizing data exist and the methods can be used for augmentation purposes where the aim is to expand the base information. When dealing with
1.1 Background

Machine learning algorithms for image processing applications, methods using simple operations such as scaling, rotation or warping are common. However, more sophisticated approaches may be used as well. Lately it has been suggested that Generative Adversarial Networks (GANs) [9] may be used for efficient data augmentation by the means of image-to-image translation. In this task the GAN is trained to transfer images between image domains. An input image is then modified by the trained algorithm to resemble the images of another domain by obtaining its characteristics, Figure 1.1 is an example. The domain translation could for example be day-to-night, summer-to-winter or one medical image modality to another. Using GANs it is possible to transfer images between domains with visually realistic results [39]. This could also prove to be a viable approach for GAN based image augmentation.

There are many application areas and image processing tasks where deep learning methods are valuable and the need for training data is accompanied by the need of data augmentation solutions. An industry that is taking advantage of the new possibilities with deep learning is the automotive industry. New driving assist features are developed continuously and for some manufacturers the long term goal is autonomous vehicles. Automotive vision systems are typically expected to handle a broad range of scenarios, including varying environments, climates, light conditions etc. It is difficult and time consuming to collect data for all situations. If a deep learning method that require training data is used, the collected data might also need to be annotated which means that even more work is required.

Another application for deep learning is the field of medical imaging. In this field the scenarios are fewer since they are limited to the different image acquisition techniques. The problem is instead that it is difficult, expensive and time consuming to recruit and scan a large number of subjects, while annotation work also requires medical knowledge. Privacy is another big issue in this field. In the case of magnetic resonance imaging (MRI), the time consumption is especially problematic since the image acquisition takes a lot more time than other imaging techniques.

In both of the mentioned cases, data synthetization using GANs could prove to be useful when simpler augmentation methods are not enough. The idea behind GANs is to enable them to improve the generation of synthetic data during training. The improvement is driven by a two-player game where the players in the game are the generator and the discriminator. The goal of the generator is to synthesize realistic data and the discriminator has the goal of distinguishing between real and synthesized data. The game ends when a Nash equilibrium is achieved. This happens when neither the discriminator nor the generator can improve without a change in the other. Training a GAN and getting it to converge to the Nash equilibrium is the main challenge when developing GANs [8]. The networks have to be synchronized so one of them does not win too easily. This would otherwise stop the weights from updating further.

Mode collapse is a known form of degenerate solution that can occur when train-
ing GANs. In this case, the generator learns to create an output that has very low variety between the generated examples. The generator fails in creating different outputs but succeeds in the goal of fooling the discriminator since the generated data is determined as realistic by the discriminator.

The performance of a GAN can depend on several factors, and the solutions to avoid issues such as mode collapse might differ. Many different types of GAN models have been proposed and are suited for different applications and datasets.

![MRI image domain translations](image1)

![Street view day and night translations](image2)

*Figure 1.1:* Translations between MR image domains and street view day and night domains using models implemented in this thesis.

### 1.2 Purpose

The purpose of this thesis is to explore the possibilities of GAN methods in creating synthetic images that are as visually realistic to the human eye as possible. Several GANs will be compared and both medical MR and street view images shall be investigated.

There is currently no golden standard how to evaluate GANs quantitatively. To achieve the goal of the thesis the evaluation needs both quantitative and qualitative tests. The evaluation will need to be designed to reveal if the generated images are visually realistic. This involves perceptual studies. For medical MR images, quantitative pixel-to-pixel measurements will be done since paired data examples are available.

The thesis will investigate if a GAN method can be used for image-to-image translation in two different applications; light and weather translations for street view
images and MR image translation between T1 and T2 weighted images. Different existing GAN models will be analyzed in order to choose two models that are to be implemented and evaluated.

The following questions are addressed in the thesis:

1. Can GANs be used to generate synthetic images from data provided by Veoneer?
2. Can GANs be used to translate between T1 and T2 weighted MR image domains?
3. How visually realistic can the synthetic images become?
4. How does the implemented GAN model perform when results are compared to ground truth data?

1.3 Motivation

GANs are deep learning generative models that are based on differentiable networks. Since its first appearance, many new GAN approaches have been proposed, with different learning routines and aims [12]. Therefore, when applying GANs to a defined task, many different factors have to be considered.

In order to obtain results, at least one GAN model needs to be implemented. The choice of model is a key decision. In this thesis, two models are chosen, implemented and evaluated. The output from different networks is dependent on several factors and there are a variety of combinations of different networks that could work for the intended applications. Several models are compared against each other to motivate the best suited methods that will be implemented.

As mentioned above, the access to data is usually critical. The same goes for the training of the GAN models investigated in this thesis. The size of the dataset depends to some extent on the complexity of the image processing task. In the case where the translation from one image to another does not require a lot of modifications to obtain a realistic result, the dataset size can be relatively small. This is the case for the grayscale MR images when compared to the street view images. However, the available MR images for this thesis are also a lot fewer than the available street view images. Understanding of needed dataset sizes and the performance impact it has is of value for all related deep learning applications.

As described earlier, data augmentation methods are interesting since the progress of deep learning to some extent depends on good utilization of accessible data. GANs are one possible direction in the field of data augmentation and thereby understanding of the limitations and possibilities is of interest. It is not guaranteed that any GAN method performs equally well when trained on data with different characteristics. The street view data provided by Veoneer differs a lot in characteristics from the medical MR data. Investigations on both datasets means
the models are evaluated in two different situations and the results will provide better understanding of generalization of the different models.

Answers to the research questions can be related to other similar image processing tasks, as well as other application areas using the chosen GAN models. If visually realistic images can be produced they can be used in other situations. For example, if annotated data exists for images in one image domain, a translation to another domain could double the annotated dataset since the annotation is also valid for the output images. The understanding of how good the results are compared to ground truth data will give knowledge as to current limitations, and future possibilities of the GAN models.

If realistic street view results are attained, Veoneer can further investigate the possibilities for using GANs in data augmentation purposes. This parallel to ongoing deep learning research for automotive assistance applications. In the case of medical images, translation from T1 to T2 images is of interest since MR is expensive and time consuming. T2 images are rare because the image acquisition takes about twice as much time compared to T1.

1.4 Delimitations

Hyperparameter values specified by the authors of the respective proposed models will be used. Only smaller changes might be made in order to get an acceptable result. Only two proposed models will be implemented in the thesis. Model modifications will however be made. No investigation on the effects of using the generated synthetic images will be done. Instead focus will be on generating as visually realistic images as possible. Quantitative evaluation will only be done on the MR images, this since paired image examples only exists for the MR dataset. The street view images will only be evaluated qualitatively.

1.5 Contributions

In this report, several GAN models are compared, implemented and evaluated. Proposed GAN models have been implemented from scratch and updated with architectural modifications and learning concepts. The models have been trained and evaluated on two different kinds of images; street view and MR images. To evaluate the result, evaluation frameworks have been developed, separately for the synthetic street view and MR images.

The results have shown great promise in many image-to-image translation tasks in terms of visual appearance of synthetic images. Exploration of possibilities on different image domain translations and on different image resolutions have resulted in valuable information to support further investigations. On MR images it has been shown that visually realistic synthetic images can be generated. On street view images, effects of different modifications such as training dataset size, identity training, image resolutions and architectural modifications on generator
and discriminator networks have been investigated.

1.6 Thesis Outline

Chapter 2 first gives some understanding of deep learning concepts which are later used to describe the different GAN models. Five different GAN based image-to-image translation models using unpaired data are then briefly explained. Related work in the medical field and GANs producing higher resolution images are also brought up.

Chapter 3 contains a comparison of the five models brought up in Chapter 2. The two models that are best suited for the thesis are then explained in more detail.

Chapter 4 describes the procedure that is used for evaluating the models. The training data is presented and explained as well as the metrics used in the quantitative evaluation. The perceptual study is also described.

Chapter 5 presents the results from the evaluation. The results from training and testing using MR data is first presented. It is then followed by the results from training and testing using street view data.

Chapter 6 discusses the results obtained from the evaluation. The method used to evaluate the different models is also discussed.

Chapter 7 concludes the study by answering the research questions presented in Section 1.2. Implications of the answers are given and suggestions to future work are presented.
2

Theory and related work

The interest of augmenting images has been seen in a variety of situations and software tools for manual editing are common. Research on image processing algorithms has allowed for automatic features such as object tracking and image enhancements. However, automated image-to-image translation has not been successful until recent years. Because of the progression of deep learning and generative models, new possibilities have appeared and new ideas and articles have emerged. This despite the fact that it so far has been difficult to find application areas for these models. Within the field of GANs, an incredible amount of new approaches have been presented in a short amount of time. Almost 300 named methods in the beginning of 2018 according to Avinash Hindupur\textsuperscript{1}. Searching for GAN related articles in Google Scholar shows an exponential rate of publications since 2014, which is another example of the quickly gained attention. This chapter introduces theory on GANs and explores prior work related to this thesis.

2.1 Convolutional neural networks

CNNs \cite{19} have become fundamental when applying deep learning algorithms on images. Similar to deep feedforward neural networks the CNNs consist of multilayer perceptrons with learnable weights and biases. The input to the network is 2D data with a depth depending on the number of channels in the image. The street view images that are used in the thesis consist of three channels and the MR images have one channel. In a layer of a neural network, the output activation of an output neuron is calculated as a linear combination of its inputs. In most cases the network performs a non-linear operation which is done by an

\textsuperscript{1}https://github.com/hindupuravinash/the-gan-zoo/blob/master/cumulative_gans.jpg
activation function. By applying a non-linear activation function the network is capable of learning more complex mappings of the data distribution since there are only linear operations in the layers.

The difference between an ordinary neural network and a CNN is the convolution that is performed by CNNs. The convolution is done by a filter that is convolved over the data and maps it to the output of the layer. More precisely, the filter is moved across the image to fixed positions, calculating an output for each position. The dimensionality of the output is the number of filters in each layer. The filter is a three dimensional matrix and the elements in the matrix are the weights that are updated when the network is training. The size of the filter is specified by the kernel size.

Common for the encoding parts of GAN models are that when increasing the number of filters or output layers, the size of the input to each layer is reduced. The reduction of each layer’s size is done by the stride, which is the step size of the filter between each convolution. A stride size of two means that the spatial dimensions of the output will be reduced to half its input size.

When training a CNN an aim must be stated so feedback can be given on how it performs. The aim is formulated as an objective function which can either be minimized or maximized depending on the wanted outcome. Based on the value from the objective function the weights and biases are updated using gradient descent with backpropagation [18]. If the objective function is minimized it can be referred to as a loss function.

### 2.2 Loss functions

The loss function is used to calculate the error of an event. An example of an event is a neural network that produces an image. The loss function could then be a resemblance measurement between the produced image and a corresponding ground truth image. There currently exists a variety of loss functions, the ones most relevant for this thesis are described below.

A loss function that is commonly used is the Mean Squared Error (MSE) defined in Equation 2.1. For example the MSE loss can be used to compare the differences in two images. The difference of the corresponding pixels in each image are calculated, squared and the mean over all pixels is calculated.

\[
L = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2
\]  

(2.1)

Another loss function that could be used to compare two images is the Mean Absolute Error (MAE) defined in Equation 2.2. A difference between the MSE loss and the MAE loss is that outliers in the MSE have a larger impact on the loss since the error is squared.
2.3 Optimizers

\[ L = \frac{1}{n} \sum_{i=1}^{n} |y^{(i)} - \hat{y}^{(i)}| \]  

(2.2)

If the aim in a neural network is to be close to a specific probability distribution, a method of measuring how different two distributions are is needed. This can be done with the Kullback-Leibler (KL) divergence [10]. The two probability distributions need to be over the same random variable. The equation for the calculation is shown in Equation 2.3. In the equation, \( P(x) \) and \( Q(x) \) are considered to be two distributions over the same variable \( x \). By using the KL as a loss function one can train the network to create probability distributions that are similar to each other.

\[ KL(P||Q) = \mathbb{E}_{x \sim P} \left[ \log \frac{P(x)}{Q(x)} \right] = \mathbb{E}_{x \sim P} [\log P(x) - \log Q(x)] \]  

(2.3)

Another method of measuring how different two distributions are is with the cross-entropy function defined in Equation 2.4. It is equivalent to KL in Equation 2.3 if \( P(x) \) does not depend on the variables that are optimized.

\[ H(P, Q) = -\mathbb{E}_{x \sim P} \log Q(x) \]  

(2.4)

2.3 Optimizers

To minimize the losses calculated from the loss functions, which are described in the previous section, gradient descent optimization algorithms are most often used. Introduced by Kingma and Ba [15] is the Adaptive Moment Estimation (Adam) optimizer which currently stands out in performance when considering computational efficiency and memory requirements. Adam is common for optimization in deep neural network applications and is used throughout this thesis.

Adam calculates learning rates that are adaptive for each parameter in its algorithm. The first and second moments, i.e. the gradient mean and variance, are estimated by using an exponentially decaying average of past gradients and squared past gradients. The two parameters, beta1 and beta2, control the exponential decay rates of the past gradients.

2.4 Residual Blocks

Image classification networks have improved their performance over the past years with deep neural networks [17] [29]. A factor that has contributed to the improved performance is the increased number of layers in the networks, which leads to deeper nets. Increasing the number of layers in a network provides additional nonlinearities which can benefit the classification task since more com-
plex solutions can be learned. With a deeper network the training becomes more complex as reported by He et al. [11]. In their article the degradation problem is addressed when creating deeper nets. When adding more layers and giving the network more parameters the performance of the network is not necessarily improved. Their solution to the problem is an architecture change, a residual block. The idea behind the residual block is that it uses an identity mapping from its input which is a shallower layer. It adds the input to the output from the layer, letting the underlying layers fit a residual mapping. The architecture is illustrated in Figure 2.1.

![Figure 2.1: Illustration of a residual block as described by He et al. [11]. The identity is added to F(x) that has been passed through the weight layers and the activation function.](image-url)

### 2.5 Variational autoencoders

Variational Autoencoders (VAEs) is a method that uses convolutional neural networks to generate data. An autoencoder can be explained as a network that learns how to compress data in a way that allows it to reconstruct it again. The purpose of the autoencoder is to reduce the dimensionality of the data, while still being able to reconstruct it with as little loss as possible. Similar to a typical autoencoder the VAE also consists of an encoder and a decoder. The aim of the VAE is however to learn the probability distribution representing the data. A data sample can then be generated by drawing a sample from the probability distribution and feeding it to the decoder.

### 2.6 Generative adversarial networks

The first method for generating synthetic data with GANs was published in 2014 by Goodfellow et al. [9]. The model created consists of two multilayer percep-
tron, a discriminator \( D \) and a generator \( G \). The purpose of the generator is to learn a data distribution \( p_g \), over the data \( x \), which it synthesizes from noise \( p(z) \). To provide feedback of how well the generator performed, the synthesized data is given to the discriminator. The discriminator estimates the probability that the provided data is generated by the generator \( p_g \). The discriminator is provided with data from both \( p_g \) and \( x \) and has the goal of maximizing the estimated probability. The generator has the opposite goal, trying to fool the discriminator into estimating the synthesized data as real. The discriminator and generator networks are trained each training iteration and are competing with each other when trying to minimize and maximize the objective in Equation 2.5.

\[
\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.5}
\]

In the beginning of the training the generator will synthesize data that will not be close to the real data distribution. It would seem as an easy task for the discriminator to classify the data as synthetic. But since the discriminator is not trained in the beginning it will be a rather challenging task to distinguish the two distributions from each other. But as more examples are presented the more experience it gets to separate the distributions. While the discriminator learns to better separate the distributions the generator learns to generate a data distribution closer to the real distribution. The performance of the network can therefore be seen as a game between the generator and discriminator where a key aspect is to make the generator and discriminator learn synchronously. If either one of the networks too easily wins over the other it will stop the learning process.

The first GAN that was published was proven on the MNIST dataset\(^2\). The generator tries to generate numbers from noise and the discriminator is trained on both real MNIST data and the synthesized data from the generator. As a result from the first publication, a cascade of new models of GANs have been developed. The new models have mainly focused on creating visually realistic images, achieving a more stable training and creating translations between different image domains.

Achieving a translation between different image domains requires a GAN that is conditional. Conditional GANs where introduced by Mirza and Osindero \([22]\) were they for example showed they could condition the output number from a model, trained on the MNIST dataset, on an input class, i.e. a number between zero and nine.

An important improvement on the original GAN model is PatchGAN. PatchGAN originates from the article Image-to-Image Translation with Conditional Adversarial Networks by Isola et al. \([13]\). The idea behind PatchGAN is that instead of having the discriminator evaluating the whole image, outputting only one value, the output is one value for a number of image patches. In the work of Isola et al. it was shown that having the discriminator to evaluate if the patch is from

\(^2\)http://yann.lecun.com/exdb/mnist/
a synthesized image instead of whole image will produce sharper image outputs from the generator.

2.7 GAN based Image-to-Image Translation using unpaired training data

The image-to-image translation task using GANs requires a conditional GAN. Unlike the first GAN mentioned in the previous section, the input to the network is an image instead of random noise. The input image is encoded and it can then be decoded or translated into the different image domains.

In the context of generative models and image-to-image translation this thesis refers to unsupervised methods as those models that do not require image pairs for training. It should not be confused with the generation of images from pure noise, as the original GAN model suggests. In the case of image-to-image translation the generated image is based on an input image, which makes the model conditional. If an image-to-image translation is supervised or unsupervised depends on if the ground truth image is included in the training. In the supervised case the translated image can be compared to the ground truth image and a loss can be generated by the comparison. In the unsupervised case the translated image is instead compared to images in the same domain but not the ground truth image.

Unsupervised data augmentation methods relying only on unpaired data have a few key advantages that make them superior for the applications of this thesis. It is hard to find and collect paired street view images of different image domains. It would require two identical image acquisitions in two different weather or light conditions, or a manual labor that pairs the images of the two datasets together in the correct manner. Since probably many different light and weather translations are wanted at some point, a method on paired data would require a lot of extra work. It is also easier to adapt or reuse an unsupervised method in future work, e.g. if synthetic data based on images from a new camera is wanted.

CycleGAN

As mentioned above, CycleGAN has proven that image-to-image translations between different types of image domains are possible. The paper by Zhu et al. [39] gained a lot of attention after publication because of the visually appealing synthetic images. The key factor for the success is the cycle-consistency criteria. Implementation wise, this criteria appears as a part in the objective function that the generator networks learn from. During training, each input image is translated to the other domain and then translated back to the initial domain. This reconstructed image is then compared to the corresponding input image and a loss is calculated using the MAE, which is described in Section 2.2. Applying this loss during training pushes the generator networks to encode the information of the input image in the translated image, to allow a good reconstruction. This reduces the space of possible learned generator mapping functions and helps the
preservation of structures in the output image.

The suggested model contains two discriminator networks and two generator networks. This means mappings between two domains, in both directions, are learned simultaneously. Many different translations with realistic results have been proven, where one is a season translation between summer and winter at the highest resolution of 256x256 pixels.

In the original CycleGAN publication [39] it is shown how the method outperforms other GANs (Bi-GAN/ALI [6, 7], CoGAN [20], SimGAN [27]) when generating synthetic images from aerial photos to maps, and vice versa. The evaluation was done by presenting synthetic or real images to humans and letting them label the images as real or fake. The best score was achieved when translating map images to aerial photos, achieving a score of 26.8%. Though it should be considered that further development on the other GAN methods have been done since the publication of the article, and the results might not be valid against the newest version of the CoGAN. A drawback is that the article has not compared its results against GAN methods such as UNIT by Liu et al. [21] or the DualGAN created by Yi et al. [37].

UNIT
The UNIT implemented by Liu et. al. [21] is a GAN implementation for image-to-image translation that uses unpaired training data. The model consists of three different spaces where two of the spaces contain a certain distribution of images such as winter, summer, night or day. The third space consists of the shared latent space of the images. The shared latent space consists of the similarities of the different domains like common underlying structures. The translation is done using variational autoencoders, generators and discriminators. The images presented in their article have visually realistic appearance but are limited to the size 480x640 pixels.

StarGAN
In most GAN models used for image-to-image translations a limitation to mappings between two image domains exists. Usually two generator networks are needed for each pair of image domains. StarGAN was developed to address this issue where a single generator supposedly can be trained to handle mappings between multiple domains [3].

To achieve this, both image input and class label for the desired image domain is sent to the generator. Training with this condition allows for full control of which target domain is wanted when later evaluating the model. To keep up with the generator the discriminator also has to learn the specific features of each image domain. To alleviate this, the discriminator also knows the domain class label during training.

GeneGAN
Another GAN method that does not require paired data is GeneGAN which was implemented by Zhou et al. [38]. In their article they demonstrate a GAN that is capable of accomplishing object transfiguration. An example of a transfiguration
that is done is changing the facial expression of a human from smiling to non-smiling. The downside of this study which would make it difficult to adapt to the street view data is that the images are spatially aligned by face landmarks.

The GeneGAN model consists of an encoder and a decoder. There are two different training sets, one set with attributes and another without attributes. From two images where one image has the attribute and the other does not have the attribute, four children are created from the two parent images. Where two of the children images are reconstructions of the parent images and the other two are the synthesized images with swapped attributes.

To achieve the attribute transfiguration an adversarial loss is introduced. Images in one domain should be distinguishable from the other domain. For example, if one domain consists of images with smiling faces and the other domain with non-smiling faces, then the model should be able to separate images in the different domains. The classification of which domain the images belong to is made by a discriminator. To create a more stable training and make sure that all information from the parent image is contained in the children, a reconstruction loss is included in the model. The reconstruction loss is applied on the reconstructed images from the different domains.

DualGAN
The DualGAN [37] has a similar architecture as the CycleGAN, which indicates that the performance of the models should be similar. Unfortunately there is no comparison between the methods. This can be due to the fact that the publication dates of their respective articles only differ with nine days. The result of the DualGAN presented by Yi et al. outperforms the cGAN [13] when translating images from day to night and sketches to photos, but not when translating labels to facades or maps to aerial photos. The evaluation was similar to the evaluation done by Zhu et al. [39], i.e. letting humans label the images, giving them a realism score. The DualGAN also lagged behind the cGAN [13] for tasks regarding semantic-based labels. A factor to consider when comparing cGAN and DualGAN is that cGAN requires paired training data while the DualGAN is an unsupervised method, it does not require paired training data. This is a possible explanation as to why the cGAN outperforms the DualGAN in most cases.

2.8 Data augmentation using GANs

It was proven by Sixt et al. [30] that GANs can be used to improve the performance of a Deep Convolutional Neural Network (DCNN) by generating more data with a RenderGAN. The aim with the article is to improve the DCNN that has the aim of tracking tagged bees. Because of different factors such as image background, lighting, object shape, position and orientation of the object the classification task increases in complexity. If the training data for the DCNN does not contain examples with different factors mentioned the performance will be poor. By using the RenderGAN, the authors generate bee tag images which are added to the training set. The DCNN is trained separately on different training sets,
with the data generated with the RenderGAN and without the RenderGAN. The result is then compared with the mean Hamming distance (MHD), i.e. the expected value of bits decoded wrong. The DCNN trained on the generated data and real data achieves a mean Hamming distance of 0.416, while the DCNN only trained on the real achieves a mean of 0.956. The work presented by Sext et al. can be related to street view images with the different factors such as rain, snow, sun, night and day are factors that affect the object classification task, similar to the factors that effect the factors effecting the tracking of bees. The variety of possible objects in street view images is however higher. For example can cars, pedestrians, trees and buildings appear in different image locations. This could potentially have a large effect making it impossible for the GAN to generate images that could be of use for improving the classification results in a street view scenario.

2.9 GANs in medical images

There are several research projects that have used GANs for medical images but there are currently no known cases where it is clinically used. An example when GANs have been used in medical images is when Wolterink et al. transferred MR images to computed tomography (CT) images [33]. Another example is transferring low-dose CT images to routine-dose images which was also done by Wolterink et al. [34]. Yang et al. recently demonstrated a transfer of T1 images to T2 images [36] inspired by the cGAN method by Isola et al. [13]. Their transfer between T1 and T2 images is the same transfer that this thesis aims to accomplish.

2.10 High resolution street view images by GANs

The synthesisization of higher resolution images requires longer training time and higher memory usage. Besides that, image-to-image translation is a more complex task than for example a common classification task since the output is an image instead of a simple classification label. Natural images of higher resolution have a higher level of detail which is expected in visually realistic synthetic images. This increases the complexity of the translation task since it places demands on the generator network when using GANs. The demand of understanding details in the images is also put on the discriminator network for it to provide feedback to the generator.

It has been shown in the work of Wang et al. [32] that a GAN can be used to create realistic high resolution street view images. They adapt a supervised translation task that uses data pairs, consisting of real street view images with the ground truth semantic label maps, to train the GAN. An interesting added feature is that they are able to manually switch between several synthetically generated instances for the different semantic label objects in the image. Using paired training data and different methods, visually realistic street view images can evidently
be generated.

Implemented in their model is also a multi-scale discriminator which later has been used in other models such as the UNIT model. The idea with the multi-scale discriminator is that downsampled versions of the image are fed to the discriminator together with the full size image. The discriminator in turn consists of similar, separate networks with different input sizes. Thereby the network with the most downsampled input has the biggest receptive field. This means the multi-scale discriminator will base its evaluation on more than one version of the input image. While utilizing the multi-scale discriminator together with Patch-GAN the patches that the discriminator will base its result on will vary in size between the original image and the downsampled image. For the downsampled image the patches will include larger parts. The discriminator will thereby have a better overview of the image.
This chapter describes the method used in this thesis. A comparison between different unsupervised image-to-image translation models using GANs is provided. Via the comparison, two models are chosen for further investigation and evaluation. These two models, CycleGAN and UNIT, are explained in depth in this chapter. The implementation procedures and model modifications are presented.

### 3.1 Comparison of unsupervised GANs

A key feature of the GAN models listed in Table 3.1 is that they are all unsupervised, i.e. they do not need paired data for training.

All presented articles in Table 3.1 show great and inspirational results in some kind of image processing task. The methods that have shown good results on image tasks close or identical to the ones this thesis aims to solve are of most interest. This because these results provide confidence for the given method. Also, one should suspect that even though results are great for some specific image processing task they might not be good for another, if some underlying assumption can not be applied to the new task. For example, StarGAN [3] shows great results of several mappings using only one generator. Here, all mappings are facial expressions and the datasets used are images of faces only, where each face is positioned in the middle of the image. This should be considered as a quite isolated situation that is not as prone to variation as street view images, where cars and pedestrians etc. might appear in many different locations in the image.

A unique feature that StarGAN has is its scalability. A single generator for multiple light and weather translations would be practical. But it is likely a lot easier to get a good result on a set of translations with less variations, like a face expres-
**Table 3.1:** The table shows the investigated unsupervised GAN methods, with relevant information for comparison between them. The number of networks says something about the total model size and complexity where G and D represent a Generator and a Discriminator respectively. Scalable means that another domain mapping can be taught without altering the main method structure. Proven on relevant translations means that the article shows successful results on any light or weather condition mapping. Image size is the image size of the output in pixels used in each related article. The given publication data is for the first publication of the related article.

<table>
<thead>
<tr>
<th>Method</th>
<th>CycleGAN</th>
<th>UNIT</th>
<th>StarGAN</th>
<th>GeneGAN</th>
<th>DualGAN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Idea</strong></td>
<td>Cycle-Consistency</td>
<td>Shared latent space between image domains</td>
<td>Multiple domain translations</td>
<td>Encoder and decoder</td>
<td>Cycle-Consistency</td>
</tr>
<tr>
<td><strong>Number of Networks</strong></td>
<td>2G + 2D</td>
<td>2VAE + 2G + 2D</td>
<td>1G + 1D</td>
<td>1Encoder + 1Decoder + 1D</td>
<td>2G + 2D</td>
</tr>
<tr>
<td><strong>Scalable</strong></td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Proven on relevant translations</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Image size</strong></td>
<td>256x256</td>
<td>640x480</td>
<td>128x128</td>
<td>178x218</td>
<td>256x256</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>Mar 2017</td>
<td>Oct 2017</td>
<td>Nov 2017</td>
<td>May 2017</td>
<td>April 2017</td>
</tr>
</tbody>
</table>

One aspect of this thesis is the image size of the output. This because high resolution images are commonly used in image processing tasks for automotive vision systems. It is more difficult to get good results on higher resolution outputs using GANs. Methods proven successful on higher resolutions are therefore of interest. None of the image sizes in Table 3.1 are high enough for state of the art automotive vision systems. Therefore methods for generating better results on higher resolutions, e.g. Progressive Growing GANs [14] by Karras et al., might be of interest in future work.

A model with more and larger networks in the architecture usually requires longer training time and more training examples. The number of networks of the different methods do not vary to a large extent, considering the case that only one domain translation is desired. Training time could be discussed when comparing the methods. However, it is difficult to estimate the respective training time and not all articles mention the training time of their results.
3.2 Image-to-Image translation using CycleGAN

The model architectures also represent the main ideas of their respective implementation. Since CycleGAN and DualGAN were developed, newer methods have adapted the notion of cycle-consistency, often implemented as a loss based on the reconstruction error using MAE. The field of GANs is quite new and popular at the time of this report, and new methods and better results appear constantly. Very often, part of the success is based on a step in the right direction by an earlier publication. All methods discussed here are from new publications, as seen by the publication dates.

As seen in the Table 3.1 CycleGAN and DualGAN are similar methods. In the respective articles they mention the other and state that similar work was being done concurrently. CycleGAN is the method mainly being referenced to by other articles. Because of the proven success of CycleGAN, on many different translation tasks, and since relevant translations for this thesis have been made on relatively high resolutions, it is chosen for implementation and further investigation. The UNIT model is also chosen, for the same reasons as for the CycleGAN model. Relevant translations have been made and especially, they have been made on high resolutions.

3.2 Image-to-Image translation using CycleGAN

The concept of the CycleGAN model comes from humans ability to picture a scene in another circumstance than the current, despite never having seen it before. For example, a person walking down a random street in a new city at day time would probably have no problem picturing what it would look like if it was night time. This despite never having been to this city before. The CycleGAN model can also learn to pick up styles, or image characteristics, from two domains, and translate examples between them. There are few restrictions as to how related the two image domains have to be, or what they should be like. The assumption made is however that the two domains have some underlying relationship. To prove the possibilities many types of translations are shown in the original article.

In the cases explored in this thesis, the underlying relationship between the two image domains is that the environment, or scene, is the same. In the medical images, the same brain should appear when translating between the T1 and T2 weighted image. In the street view images objects like cars, road markings and pedestrians, that are present in the input image, should also appear in the output image. This thesis only investigates a type of style transfer in some sense, but it should be noted that some translations might allow the model to freely come up with ideas on how to render an environment. A translation that is discussed in this report is the one from night to day. For example, a street view image taken during night-time could contain some buildings and trees far ahead, not visible in the image but that would be visible in a corresponding day image.

The goal when training the networks is that the outputs are indistinguishable from the true images in the respective domains. The two image domains are
called $A$ and $B$, where real images $a \in A$ and real images $b \in B$. The generator that is trained to translate an image from domain $A$ to $B$ is called $G_{A2B}$, and the opposite generator $G_{B2A}$. A synthetic image in domain $B$ is called $\hat{b} = G_{A2B}(a)$, and a synthetic image in domain $A$ $\hat{a} = G_{B2A}(b)$. Indistinguishable means in this sense that the distribution over $\hat{b}$ matches the empirical distribution over $p_{data}(b)$. A problem with only matching distributions is that an output image that has the right distribution will fulfill this requirement regardless of the input. CycleGAN addresses this mode collapse problem by the cycle-consistency criteria where the two generators are trained to be the inverse of each other. During training the input in one domain is translated to the other domain. It is then translated back, resulting in a reconstructed image. The reconstructed image should look like the input image, i.e. $G_{B2A}(G_{A2B}(a)) \approx a$.

### 3.2.1 Training

The diagram in Figure 3.1 shows how the CycleGAN operates. A simultaneous process for translations in the other direction also occurs during training. When training the generators, an input from domain $A$, $a$, is translated by the generator $G_{A2B}$, into a synthetic image $\hat{b}$. This synthetic image is passed to the discriminator in domain $B$, $D_B$, where it is evaluated. An adversarial loss similar to the one used in the first proposed GAN model, given by Equation 2.5, is also used during training. The adversarial loss for the $G_{A2B}$ is calculated according to Equation 3.1, where a lower loss is given if the synthetic image fools the discriminator into outputting a value close or equal to one.

$$L_{G_{A2B}}(G_{A2B}, D_B, a) = \mathbb{E}_{a \sim p_{data}}[(D_B(G_{A2B}(a)) - 1)^2]$$  \hspace{1cm} (3.1)

In the opposite manner, the discriminator tries to minimize the output on synthetic images while instead maximizing the probability for real images. This means minimizing Equation 3.2.

$$L_{D_B}(D_B, a, b) = \mathbb{E}_{a \sim p_{data}(a)}[(D_B(G_{A2B}(a)))^2] + \mathbb{E}_{b \sim p_{data}(b)}[(D_B(b) - 1)^2]$$  \hspace{1cm} (3.2)

The discriminator learns from training on both real and synthetic images with labels as ones and zeros respectively. This is shown in Figure 3.2 and corresponds to minimizing Equation 3.2. The adversarial loss for the $A2B$ translation can be summarized according to Equation 3.3. A corresponding loss is applied for the translation from domain $B$ to domain $A$.

$$L_{GAN}(G_{A2B}, D_B, a, b) = L_{D_B}(D_B, b) + \min_{G_{A2B}} \max_{D_B} L_{G_{A2B}}(G_{A2B}, D_B, a)$$  \hspace{1cm} (3.3)

The generator learns by minimizing the adversarial loss in Equation 3.3. It also learns from a cyclic loss. The cyclic loss is calculated as the MAE, explained in
3.2 Image-to-Image translation using CycleGAN

**Figure 3.1:** Generator training for generator $A2B$, ‘Gen $A2B$’. Real images in domain $A$, ‘Real $a$’, flow according to the diagram. The generator weight update depends on the cyclic loss and the output from discriminator in domain $B$, ‘Dis $B$’. The same flow, in the opposite direction, goes for real images from domain $B$. Both generators are trained on the cyclic losses calculated from both directions.

**Figure 3.2:** Discriminator training for discriminator in domain $B$, ‘Dis $B$’. Discriminator weights are updated depending on the output from real and synthetic images. Real images have ones as labels, synthetic images have zeros. The discriminator in domain $A$ is trained in the same manner, on real and synthetic images in domain $A$. 
Section 2.2, between the input image and the reconstructed image. Both generators are trained based on both cyclic losses, shown in Equation 3.4, each training iteration.

\[
L_{cyc}(G_{A2B}, G_{B2A}) = \mathbb{E}_{a \sim p_{data}(a)}[\|G_{B2A}(G_{A2B}(a)) - a\|_1] + \\
\mathbb{E}_{b \sim p_{data}(b)}[\|G_{A2B}(G_{B2A}(b)) - b\|_1]
\]

(3.4)

A hyperparameter, \(\lambda\), is introduced and controls the impact of the cyclic loss. The full objective function of the CycleGAN model results in Equation 3.5 below.

\[
L(G_{A2B}, G_{B2A}, D_A, D_B) = L_{GAN}(G_{A2B}, D_B, a, b) + \\
L_{GAN}(G_{B2A}, D_A, b, a) + \\
\lambda L_{cyc}(G_{A2B}, G_{B2A})
\]

(3.5)

### 3.2.2 Implementation

Appertained to the CycleGAN article [39] is an open github repository\(^1\) that contains the original implementation done in PyTorch [25]. Several other implementations are referenced to from the repository. In this thesis, the CycleGAN model is implemented from scratch in Keras [4] with a Tensorflow [1] backend. Table 3.2 shows the network architectures.

According to the article the last layer of the generator architecture contains a Rectified linear unit (ReLU) activation function, but the corresponding code use tanh instead and is therefore used in this thesis. The suggested model settings in the article are:

- **Batch size** = 1
- **Learning rate** = \(2 \times 10^{-4}\), Linear decay from initial value to zero over the last 100 epochs
- **lambda** = 10.0
- **beta_1** = 0.5, for the Adam optimizer
- **beta_2** = 0.999, for the Adam optimizer

The model using these is referred to as the CycleGAN baseline in this report. The article also discusses some modification to the baseline implementation.

- **Update discriminators on history of synthetic images** - Adapted from Shrivastava et al. [28] the discriminators are updated using a history of 50 synthetic images in each image domain. In each training iteration there is a 50% probability that the most recent batch of synthetic images are used, otherwise the batch replaces random synthetic images from the buffer and

\(^1\)https://github.com/junyanz/CycleGAN
Table 3.2: The original CycleGAN model for 256x256 images. The discriminator networks are 70x70 PatchGAN networks [13]. Each residual block contains two 3x3 convolutional layers with 128 filters on both layers. After the last layer of the discriminators, a convolution is applied to produce a one dimensional output, i.e. the discriminators’ estimation on the realness of the input image. See Sections 2.1 and 2.4 for details about convolutional layers and residual blocks.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Generators</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Convolutional-(Filters-32, Kernel size-7, Stride-1), BatchNorm, ReLU</td>
</tr>
<tr>
<td>2</td>
<td>Convolutional-(Filters-64, Kernel size-3, Stride-2), BatchNorm, ReLU</td>
</tr>
<tr>
<td>3</td>
<td>Convolutional-(Filters-128, Kernel size-3, Stride-2), BatchNorm, ReLU</td>
</tr>
<tr>
<td>4−12</td>
<td>Residual block-(Filters-128, Kernel size-3, Stride-1), BatchNorm, ReLU</td>
</tr>
<tr>
<td>13</td>
<td>Convolutional-(Filters-64, Kernel size-3, Stride-0.5), Fractionally strided, BatchNorm, ReLU</td>
</tr>
<tr>
<td>14</td>
<td>Convolutional-(Filters-32, Kernel size-3, Stride-0.5), Fractionally strided, BatchNorm, ReLU</td>
</tr>
<tr>
<td>15</td>
<td>Convolutional-(Filters-3, Kernel size-7, Stride-1), BatchNorm, Tanh</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Discriminators</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Convolutional-(Filters-64, Kernel size-4, Stride-2), LeakyReLU with slope 0.2</td>
</tr>
<tr>
<td>2</td>
<td>Convolutional-(Filters-128, Kernel size-4, Stride-2), InstanceNorm, LeakyReLU with slope 0.2</td>
</tr>
<tr>
<td>3</td>
<td>Convolutional-(Filters-256, Kernel size-4, Stride-2), InstanceNorm, LeakyReLU with slope 0.2</td>
</tr>
<tr>
<td>4</td>
<td>Convolutional-(Filters-512, Kernel size-4, Stride-2), InstanceNorm, LeakyReLU with slope 0.2</td>
</tr>
</tbody>
</table>
those from the buffer are used to train the discriminators. This supposedly reduces model oscillation.

- **Identity loss** - Each generator is trained on images from the domain the generator is supposed to translate input images to. This to teach it to identity map images that already have the characteristics of the target domain.

These updates are included in the Keras implementation. For investigation purposes, some other model updates, or options, are also implemented.

- **PatchGAN removal** - Instead of allowing the discriminators to output a value for each patch, an option to train using discriminators with only one output was implemented.

- **Multi-scale discriminator** - Adapted from Wang et al. [32] an option for using a multi-scale discriminator was implemented. The implemented multi-scale discriminator contains two, instead of three as suggested by Wang, original discriminators. The inputs to the discriminator are different versions of the input image. The first is the full size and the second is a downsampled version of the input image, resized with a factor two.

- **Supervised learning** - An option to allow supervised learning is implemented where the supervised loss is calculated by the MAE between the output and ground truth data.

- **Expanded generator architecture** - To allow the generators the same encoding as the second multi-scale discriminator, an option to increase the generator architecture with an extra encoding step, up to 256 filters, before the residual blocks, is implemented. This is followed by an added fractionally strided convolution, i.e. deconvolution, layer after the residual blocks.

- **Resize convolution** - To handle checkerboard pattern artifacts, an option for using resize convolution [24] is implemented. Resize convolution is done by an upsampling followed by a convolution layer and the implementation was done by an upsampling of a factor two, followed by a reflection padding before a 3x3 convolution with stride one.

- **Dynamic image resolution** - The original article only shows results for square images. To allow testing and evaluation on different image sizes the architectures are implemented to work independently of image resolution.

Results from different combinations of the mentioned options are presented in Chapter 5 and discussed in Chapter 6.

### 3.3 Image-to-Image translation using UNIT

One of the chosen models to implement is the one created by Liu et al. [21]. The model includes variational autoencoders, weight sharing layers, generators, discriminators and cycle-consistency. An assumption is made in the model where images in two different domains, domain A and domain B share a latent space Z.
In the latent space the images exist in the same space, meaning that the images can be translated to the different domains as shown in Figure 3.3.

**Figure 3.3:** A simplified illustration of the translation and reconstruction between the different domains in the UNIT model. The images in the different domains A and B can be translated to the opposite domain via Z.

The different domains could for example be night and day street view images. The two real images, \( a \in A \) and \( b \in B \) are mapped to the shared latent space by first passing through two separate encoders \( E_A(a) = a_e, E_B(b) = b_e \) and then passing the outputs through a shared encoder \( E_Z(a_e) = a_z, E_Z(b_e) = b_z \). After the images have been passed through the shared encoder the reparametrization trick is utilized [16], which is further described below, where \( a_z, b_z \) become \( z_A, z_B \). The \( z_A \) and \( z_B \) are data representations of the images in the shared latent space. From the shared latent space both of the original images can be recovered and also translated into the different domains. This is done by passing the data through a shared decoder \( D_E Z(z_A) = a_d, D_E Z(z_B) = b_d \). Depending on the wanted output domain the images are passed through different generators which synthesize the encoded images \( G_A(a_d) = a_\hat{a}, G_A(b_d) = b_\hat{a}, G_B(a_d) = a_\hat{b}, G_B(b_d) = b_\hat{b} \). The synthesized images are thereafter passed through different discriminators depending on which domain the translated images are in, \( D_A(b_\hat{a}) \) or \( D_B(a_\hat{b}) \). Notable is that it is only the images that have been translated that are passed to the discriminators. The discriminators evaluate the synthetic images and a loss is calculated based on the discriminators’ evaluation and the true label. The true label is one for real images and zero for synthetic images. The model with its different translation and reconstruction flows is illustrated in Figure 3.4.
Figure 3.4: Illustration of the translation and reconstruction in the UNIT model. In the shared latent space the two images are represented as $a_z$ and $b_z$ which are both passed through the shared decoder $DE_Z$ and the different generators. The output from $G_A$ is an image from domain B which is now in domain A and vice versa for $G_B$.

The shared latent space assumption made in the model also includes a cycle-consistency constraint that is similar to the one explained in the CycleGAN model in Section 3.2. Meaning that the synthesized image can be reconstructed to the original image. The shared latent space assumption includes the cycle-consistency constraint but the cycle-consistency constraint does not imply the shared latent space assumption.

3.3.1 Model Framework

The model consists of eight different subnetworks which are represented as rectangles in Figure 3.4. Three of them serves as encoders, $E_A$, $E_B$ and $E_Z$, one is a decoder $DE_Z$, two are generators, $G_A$, $G_B$ and two are discriminators $D_A$, $D_B$. Different combinations of the subnetworks have different roles which are described below.

3.3.2 Variational autoencoder

There are two VAEs [5] in the model, $VAE_A$ and $VAE_B$. Where $VAE_A$ in domain A consists of $E_A$, $E_Z$, $DE_Z$ and $G_A$. The second VAE, $VAE_B$ in domain B, consists of $E_B$, $E_Z$, $DE_Z$ and $G_B$.

The purpose of using VAEs is that they can learn complex data distributions of different image domains. The VAEs first encode the real images in the different encoders, $E_A(a) = a_c$ and $E_B(b) = b_c$, and then pass the output from the encoders to the shared encoder which gives the outputs $E_Z(a_c) = a_z$ and $E_Z(b_c) = b_z$ in the shared latent space. The components in the shared latent space $Z$ are assumed in the model to be conditionally independent and Gaussian with unit variance. The
distribution of the latent code $z_A$ is given by $q_A(z_A|x_A) \equiv \mathcal{N}(z_A|a_z, I)$ where $I$ is the identity matrix.

To be able to train the network with backpropagation and gradient descent the reparametrization trick [16] is used. The reparametrization trick can be implemented by adding noise to the outputs from the shared encoder $E_Z(a_e) = a_z$ and $E_Z(b_e) = b_z$, which produces the outputs $z_A = a_z + \eta$ and $z_B = b_z + \eta$. The added noise has a Gaussian distribution, $\eta \sim \mathcal{N}(\eta|0, I)$, where $I$ is an identity matrix. If the data had a distribution of a Dirac delta function instead of Gaussian, the gradient descent would not be possible.

### 3.3.3 Weight-sharing layers

The images in the different domains $A$ and $B$ are assumed, from the shared latent space assumption, to have a common high level scene representation. Meaning that even though a street image is in the night domain the same image will still have the same features such as a car, street or tree in the day domain. This assumption is used by sharing the weights for both domains in the shared encoder $E_Z$ and the shared decoder $D_{eZ}$. The low level representations are not shared and are represented by $E_A, E_B, G_A, G_B$. In a street view image the low level representations could for example be that the trees are dark in the night domain and bright in the day domain.

### 3.3.4 GAN components

The model consists of two GANs, which are further described in Section 2.6, where the first, $GAN_A$, consists of the components $G_A$ and $D_A$ and the second, $GAN_B$, consists of $G_B$ and $D_B$. Similar to traditional GANs the generator synthesizes an image that the discriminator evaluates as real or synthesized. Real images are also given to the discriminator and similar to the synthesized images the discriminator evaluates the real images as real or synthesized. The feedback for the generator is based on a binary cross entropy loss, described in Section 2.2. It is based on the evaluation from the discriminator and the corresponding labels to the images. The discriminator itself is trained on the loss calculated from its evaluation of the real and synthesized image.

The output from $G_A$ and $G_B$ are two images each, where one is the translated image and the other is the reconstructed image. The images that are passed to the discriminators are the synthesized images, $a\hat{b}$ and $b\hat{a}$ and not the reconstructed images $a\hat{a}$ and $b\hat{b}$.

### 3.3.5 Cycle-consistency

As earlier mentioned, the model includes a cycle-consistency. The images that are translated back to their original domains are the translated images $ab$ and $ba$. The cycle is illustrated in Figure 3.5.
Figure 3.5: Illustration of the cycle-consistency in the UNIT model. The synthesized images $\hat{a}b$ and $b\hat{a}$ are passed through the network to create the reconstructed images $\hat{a}b\hat{a}$ and $b\hat{b}$ that are passed to the discriminators in each domain.

### 3.3.6 Learning

The combined learning problem for all components $\text{VAE}_A$, $\text{VAE}_B$, $\text{GAN}_A$, $\text{GAN}_B$ and cyclic loss ($\text{CC}_{AB}$) has been summarized in Equation 3.6. To note is that $E_Z$ and $DE_Z$ are part of all components and have been excluded for the readability.

$$
\min_{(E_A,E_B,G_A,G_B)} \max_{(D_A,D_B)} \mathcal{L}_{\text{VAE}_A}(E_A,G_A) + \mathcal{L}_{\text{VAE}_B}(E_B,G_B) + \\
\mathcal{L}_{\text{GAN}_A}(E_A,G_A,D_A) + \mathcal{L}_{\text{GAN}_B}(E_B,G_B,D_B) + \\
\mathcal{L}_{\text{CC}_{AB}}(E_A,G_A,G_B) + \mathcal{L}_{\text{CC}_{BA}}(E_B,G_B,G_A)
$$

Equation 3.6

The training of the VAEs aims to minimize the functions in Equations 3.7 and 3.8. The prior distribution, $p_\eta(z) = \mathcal{N}(z|0, I)$, and the KL divergence, described in Section 2.2, are used to push the distribution of the shared latent space data towards a Gaussian distribution. The hyperparameters $\lambda_1$ and $\lambda_2$ are weights to control the impact of each function. The second part of the equations has the purpose of minimizing the distance between the reconstructed image and the real image.

$$
\mathcal{L}_{\text{VAE}_A}(E_A,G_A) = \lambda_1 \text{KL}(q_A(z_A|a)\|p_\eta(z)) + \\
\lambda_2 \mathbb{E}_{z_A \sim q_A(z_A|a)}[\log p_G(a|z_A)]
$$

Equation 3.7

$$
\mathcal{L}_{\text{VAE}_B}(E_B,G_B) = \lambda_1 \text{KL}(q_B(z_B|b)\|p_\eta(z)) + \\
\lambda_2 \mathbb{E}_{z_B \sim q_B(z_B|b)}[\log p_G(b|z_B)]
$$

Equation 3.8
The GAN functions in Equation 3.6 are described in Equations 3.9 and 3.10. The first term in the equation takes in a real image and the discriminator outputs a value depending on the evaluation of the image. In the second term of the equation the discriminator takes in a synthesized image that has been translated to the domain. The hyperparameter $\lambda_0$ determines the impact of the GAN loss.

\[
L_{\text{GAN}}(E_A, G_A, D_A) = \lambda_0 \mathbb{E}_{a \sim P_a} [\log D_A(a)] + \\
\lambda_0 \mathbb{E}_{z_B \sim q_B(z_B|b)} [\log(1 - D_A(G_B(z_B)))]
\]

\[
L_{\text{GAN}}(E_B, G_B, D_B) = \lambda_0 \mathbb{E}_{b \sim P_b} [\log D_B(b)] + \\
\lambda_0 \mathbb{E}_{z_A \sim q_A(z_A|a)} [\log(1 - D_B(G_A(z_A)))]
\]

The cycle-consistency constraint is modeled according to Equations 3.11 and 3.12. The equation contains two different KL divergence terms where the first is similar to the one in Equations 3.7 and 3.8. The second KL divergence term is for the reconstructed image. The third term of the equation is based on how similar the reconstructed images are to the real images in their respective domains. The hyperparameters $\lambda_3$ and $\lambda_4$ determine the impact of the cyclic loss.

\[
L_{\text{CCA}}(E_A, G_A, E_B, G_B) = \lambda_3 KL(q_A(z_A|a)\|p_\eta(z)) + \\
\lambda_3 KL(q_B(z_B|a\hat{b})\|p_\eta(z)) - \lambda_4 \mathbb{E}_{z_B \sim q_B(z_B|a\hat{b})} [\log p_G(A|z_B)]
\]

\[
L_{\text{CCB}}(E_B, G_B, E_A, G_A) = \lambda_3 KL(q_B(z_B|b\hat{a})\|p_\eta(z)) + \\
\lambda_3 KL(q_A(z_A|b\hat{a})\|p_\eta(z)) - \lambda_4 \mathbb{E}_{z_A \sim q_A(z_A|b\hat{a})} [\log p_G(B|z_A)]
\]

Training on Equation 3.6 results in a classic GAN game between the discriminators and the encoders, decoders and generators. As in all GANs the training is a balance. The competitors should be on a similar level for the training not to end because of for example mode collapse. One way of achieving this is to trim the hyperparameters for the model. Another way is to change the architecture of the model. The authors of the article describing the UNIT model show that they have done these adjustments and tests when creating the model.

### 3.3.7 Implementation

The original implementation of the UNIT model is available from the open repository on github\(^2\) where the model is implemented in PyTorch [25]. In the repository there are different architectures of the model. The one that is implemented in this thesis is the one created for day to night conversion of street view images. The architecture of the model can be seen in Table 3.3. The implementation of the model in this thesis is referred to as the VAE-CoupledGAN model and is done from scratch in Keras [4] with a Tensorflow [1] backend. The suggested model set-

---

\(^2\)https://github.com/mingyuliutw/UNIT
Settings from the UNIT model are used in the Keras implementation, and are the following:

- **Batch size** = 1
- **Learning rate** = $1 \times 10^{-4}$
- lambda 0 = 10
- lambda 1 = 0.1
- lambda 2 = 100
- lambda 3 = 0.1
- lambda 4 = 100
- beta_1 = 0.5, for the Adam optimizer
- beta_2 = 0.999, for the Adam optimizer

The model using these settings is referred to as the VAE-CoupledGAN baseline. The official repository of the UNIT model has two modifications that are not mentioned in the article which were implemented in the Keras model; patchGAN and multi-scale discriminator.

**PatchGAN and Multi-scale Discriminator**

What is not mentioned in the UNIT article but is implemented in the official method on Github is a PatchGAN architecture in the discriminator. Another important feature that is not mentioned in the article but is implemented in the official method on Github is the multi-scale discriminator.

- **PatchGAN** - Similar to the one implemented in the article Image-to-Image Translation with Conditional Adversarial Networks by Isola et al. [13]. The discriminator evaluates areas of a synthesized image, so called patches, instead of the whole image.

- **Multi-scale discriminator** - Inspired by the article produced by Wang et al. [32] the discriminator takes three inputs instead of one. Two of the inputs are downsampled with a factor of two and four.
3.3 Image-to-Image translation using UNIT

Table 3.3: The table describes the UNIT model architecture for day to night conversion which is used in the Keras implementation of the VAE-CoupledGAN model. Each layer is described with the number of filters, kernel size, stride size and the activation function. There are two weight sharing layers, meaning that the data from the different domains goes through the same layer. The convolutional layers and the residual blocks are described in Section 2.1 and Section 2.4 respectively.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Encoders</th>
<th>Shared layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Convolutional-(Filters-64, Kernel size-7, Stride-1), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Convolutional-(Filters-128, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Convolutional-(Filters-256, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Residual block-(Filters-512, Kernel size-1, Stride-1)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Generators</th>
<th>Shared layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Residual block-(Filters-512, Kernel size-3, Stride-1)</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Deconvolution-(Filters-128, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>Deconvolution-(Filters-64, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Deconvolution-(Filters-3, Kernel size-1, Stride-1), Tanh</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Discriminators</th>
<th>Shared layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Convolutional-(Filters-64, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Convolutional-(Filters-128, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Convolutional-(Filters-256, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Convolutional-(Filters-512, Kernel size-3, Stride-2), LeakyReLU</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Convolutional-(Filters-1, Kernel size-1, Stride-1), Sigmoid</td>
<td>No</td>
</tr>
</tbody>
</table>
The choice of the two models to implement, CycleGAN [39] and UNIT [21], is based on the comparison study in the theory chapter. Both models show promising results on light and weather translations. Both methods take a reconstruction error into consideration but there are more assumptions made on the image domains when using the UNIT method. If the assumptions are valid or not is difficult to determine before implementation and testing, but this difference between the two methods make them an interesting pair to evaluate and compare.

4.1 Data

The data used for evaluation is divided into training and test sets for street view and medical images. The data in each case is also from two separate image domains.

4.1.1 MRI data

The dataset used in the evaluation is provided by the Human Connectome project\(^1\). It consists of axial images with segmented brains from the center slice (slice number 120). The dataset has paired data for T1 and T2 weighted images from 1113 patients. All of the images have been fitted to a common template so they are in the same position and size. The image acquisition was done using the Siemens Connectome Skyra 3T MR scanner [31] [23].

---

\(^1\)Human Connectome Project, WU-Minn Consortium (Principal Investigators: David Van Essen and Kamil Ugurbil; 1U54MH091657) funded by the 16 NIH Institutes and Centers that support the NIH Blueprint for Neuroscience Research; and by the McDonnell Center for Systems Neuroscience at Washington University.
4 Evaluation Procedure

Figure 4.1: Examples of MR medical images. A T1 and a T2 weighted MR image is shown in (a) and (b) respectively.

The data is split up into a training set of 900 images in each domain. These are used to train all models in the MR evaluation. The remaining 213 images, in each domain, are used for testing. All quantitative results in this report are based on this test dataset. Figure 4.1 shows examples of MR medical images.

4.1.2 Street view data

The street view datasets were provided by Veoneer. All comparisons between models and model settings are done on translation results between night and day. In addition, other translations were tried out. The small dataset consists of images from city environment with a clear sky. The training dataset has 1508 day and 1999 night images and the test dataset 447 day and 660 night images. A larger dataset is also created. The large dataset contains 380,000 day and night images, i.e. 760,000 images in total, where any road type, sky and environment is allowed. Models trained on this dataset are compared to models trained on a subset of the large dataset. The subset contains 1900 images in each domain and is derived by taking random images from the larger dataset, to match the distribution of street view scenarios. The test dataset corresponding to the larger dataset, and thereby the subset, consists of 500 day and night images.

A dataset containing images of sunny and rainy weather is also used. It contains 2656 sunny and 1907 rainy street view images for training, together with 25 test images in each image domain.

4.2 Synthetic MR data evaluation

The aim of the evaluation is to investigate the differences between the methods used to generate synthetic images. It is also interesting to compare the results
4.2 Synthetic MR data evaluation

**Table 4.1: Models used in the MR evaluation.**

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
<th>Training details</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG_bl</td>
<td>CycleGAN baseline implementation in Keras with settings according to suggestions from the original article. See Section 3.2.2 for details.</td>
<td>175 epochs</td>
</tr>
<tr>
<td>G_s</td>
<td>CycleGAN generators trained only supervised, i.e. by loss from MAE between output and ground truth data.</td>
<td>200 epochs</td>
</tr>
<tr>
<td>CG_s</td>
<td>CycleGAN baseline with added supervised loss as in G_s. The supervised loss weight was set equal to the cyclic loss weight, i.e. 10.0.</td>
<td>125 epochs</td>
</tr>
<tr>
<td>VAE-C</td>
<td>VAE-CoupledGAN baseline implementation in Keras with settings according to suggestions from the original article. See Section 3.3.7 for details.</td>
<td>200 epochs</td>
</tr>
<tr>
<td>UNIT</td>
<td>The original UNIT implementation in Pytorch. Settings according to suggestions from the original article.</td>
<td>200 epochs</td>
</tr>
<tr>
<td>Simple</td>
<td>Simple network consisting of: Convolutional-(Filters-256, Kernel size-1, Stride-1) ReLU Convolutional-(Filters-1, Kernel size-1, Stride-1)</td>
<td>200 epochs</td>
</tr>
</tbody>
</table>

against supervised methods using both a simple and a complex model. This is to quantify the effects of supervised training compared to adversarial training while also visually inspecting the differences in the synthesized images.

The models compared in the evaluation are shown in Table 4.1. The goal of all models were to train them for 200 epochs. This was accomplished with half of the selected models, G_s, VAE-C, UNIT and Simple. The CG_bl and CG_s were more unstable and after training 200 epochs there were artifacts in the synthetic images. The models chosen for the evaluation where therefore selected when no artifacts were visible in the synthetic images. This resulted in models that were trained between 125 and 200 epochs.

The original CycleGAN implementation in PyTorch is not part of the evaluation, despite the presence of the original UNIT implementation in PyTorch. This is because of the complexity of the UNIT model where an implementation in a new deep learning framework, Keras in this case, is more complex. This because of
backend differences between PyTorch and Keras which introduces a possibility to more errors in the implementation. The Keras implementation of the CycleGAN was instead verified by training on the Yosemite dataset from the original article where visually similar results were attained.

It should also be noted that the UNIT model differs from all models in two aspects when loading the training data. The first is that it does not divide each image with the value of 255, which is the maximum pixel value. Instead it divides each image with the maximum pixel value in the corresponding image. The second aspect is a data augmentation method. It is a 50 % chance that the training image is horizontally flipped before processing the image.

The Simple model is part of the evaluation to act as a baseline. Through the architecture of the Simple model it is only allowed to learn an intensity mapping. In the case of translations between T1 and T2 images this is not enough for a great results, but will suffice as a baseline.

### 4.2.1 Quantitative evaluation

All trained models are used to generate synthetic images based on all images in the MR test dataset. Each image is compared to its corresponding ground truth image. The average result, over all images, represents the models performance, T1 and T2 images separately. Since the MR images can naturally differ in intensity, each image is normalized before the calculations by division of the standard deviation and subtraction of the mean value.

The evaluation metrics are adapted from Yang et al. [36] where three measurements are used; MAE, peak signal noise ratio (PSNR) and mutual information (MI). The MAE is calculated as the mean of the absolute error between each pixel value in the ground truth and synthetic image. The PSNR is defined in Equation 4.1 where MAX indicates the highest pixel value in the ground truth and synthetic image pair, and MSE is calculated between each pixel value in the ground truth and synthetic image, as described in Section 2.2. This means the MSE acts as the noise in the calculation.

\[
PSNR = 10 \log_{10} \frac{\text{MAX}^2}{\text{MSE}} \tag{4.1}
\]

The MI is calculated according to Equation 4.2 where \(a_i\) and \(\hat{a}_i\) are the pixel values of the ground truth and synthetic image respectively. The marginal densities of the ground truth and synthetic image are denoted as \(p(a_i)\) and \(p(\hat{a}_i)\), and \(p(a_i, \hat{a}_i)\) is the joint probability density.

\[
MI(a, \hat{a}) = \sum_{a_i \in a} \sum_{\hat{a}_i \in \hat{a}} p(a_i, \hat{a}_i) \log \left( \frac{p(a_i, \hat{a}_i)}{p(a_i)p(\hat{a}_i)} \right) \tag{4.2}
\]

The MAE between two images does not reveal anything about similarities in struc-
tures or image quality. However, a large error means the results differ a lot from the original image. The MAE can thereby be used together with the other metrics to evaluate the synthetic images.

PSNR can be used to measure quality after encoding/decoding losses [26], and since image translation between domains resemble this, PSNR is used in this report.

If considering two images as random variables, MI calculates what the name implies. A high MI score indicates that a lot can be said about one of the images by observing the other image. The score is calculated by creating a 2D histogram over two images. The histogram is divided into bins which were chosen to 125 [2]. If the two images has a high number of pixel values located in the same bin of the 2D histogram, a higher MI value will be given. If the pixel values are located in more bins a lower MI value will be given.

### 4.2.2 Qualitative evaluation

To visually evaluate a synthetic image compared to a real image can be difficult if the differences are small. A solution to the visual inspection is to instead visualize a relative error between the real image and the synthetic image. This is done by calculating the absolute difference between the images divided by the real image. These calculations are done on images normalized in the same manner as for the quantitative evaluation and the error is the relative absolute difference.

The perceptual study is done by the associate professor Anders Eklund who has several years of experience working with MR images. The professor has worked with both T1 and T2 images but has more experience with T1 images. He receives T1 and T2 images where half of them are real and the other half consists of synthetic images from the models in Table 4.1. The synthetic set are an equal number of images from each model, in both domains, added up to 96 synthetic images, 48 T1 and 48 T2 images. The set of real images are also 48 T1 and 48 T2 images. The models that were used in the perceptual test and the number of images used for the different models can be see in Table 4.2. The $G_s$ and Simple are not evaluated since it is obvious that the images are synthetic, this because of the smoothness in the images.

All images are numbered with a unique and random number. The first 96 images are real and synthetic T1 and the rest are T2 images. All images are visually inspected one by one and labeled as real or synthetic in a text document. A corresponding text document with the true labels is used to calculate the number of correct and incorrect labels for each model, as well as the real images. Going back and changing a labeled image is not allowed.

For all the real images, synthetic images and for each model the number of correct labels and incorrect labels is calculated which provides a quantitative measurement to compare the models’ ability to generate visually realistic images. To complement the results, the synthetic images from the different models are visually inspected more thoroughly and comments on the visual appearances are
Table 4.2: Number of images from the different models and real images that were used in the perceptual test.

<table>
<thead>
<tr>
<th>Images</th>
<th>T1</th>
<th>T2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>CG_bl</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>CG_s</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>VAE-C</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>UNIT</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

given by the expert.

4.3 Synthetic street view data evaluation

Several models, with different model settings and adjustments in architecture, are to be compared in the evaluation on the street view data. Each model that is evaluated is trained for at least 100 epochs. To simplify the evaluation, by ruling out the bad performing models early, results are printed during training. This means only the models that perform as hoped are fully trained, i.e. at least 100 epochs, and later used in the evaluation. The training is done on the street view, day and night dataset.

The available street view data has no image pairs which affects the possibilities of the evaluation. The results from the models trained using the street view datasets will thereby not be evaluated in a quantitative manner. All synthetic images are generated using images in the street view test datasets. When comparing two models against each other the five most visually realistic images and the five least visually realistic images are chosen from the synthetic images. The evaluation of the models is then based on the chosen images. For every pair of models that are compared, a discussion is done. The factors that are considered are:

- General appearance.
- Level of details - edges, cars, etc.
- Preservation of colors on objects - road lines, cars, road signs, etc.
- Preservation of lines - road lines, houses, etc.
- Light adjustments - car lights, street lights, etc.

The initial comparison between the CycleGAN and VAE-CoupledGAN model is
4.3 Synthetic street view data evaluation

done on images of size 256x256 pixels.

4.3.1 Training with identity mapping

In the case of street view images, identity mapping can help handling special cases like tunnels. Street view images inside tunnels look similar to regular night images. This means a trained generator for night to day translation will try and make a day version of the tunnel, when the correct behaviour would be to keep the tunnel appearance.

To evaluate the effect of identity training, the CycleGAN baseline model is updated with identity mapping at every n:th training iteration. The variable \( n \) determines how often the identity training is done, e.g. \( n = 100 \) means every hundred training iteration an identity mapping is done, and the weights are updated accordingly.

The identity training involves a night image being passed to the day-to-night generator, and weights are updated based on an MAE loss between input and output. The night-to-day generator is updated similarly, where the input is a day image.

4.3.2 Higher resolution images

The models are also compared on higher resolution, i.e. 512x512 pixels. The comparison is done on results using the small street view, day-night dataset. The CycleGAN model with settings according to the article is compared against CycleGAN models with and without expanded generator architectures, multi-scale discriminators and resize convolution. Since the VAE-CoupledGAN model has been proven to give good results on resolutions up to 640x480 pixels, it is part of this evaluation. The VAE-CoupledGAN model is using the suggested settings in the original article. All compared models are trained on the small day and night street view image dataset for 100 epochs.

4.3.3 Larger dataset

The access to a lot of images, provided by Veoneer, is used in this thesis to evaluate the effect of a larger dataset by comparing results in translations between night and day street view image domains. The evaluation is done using the CycleGAN baseline model, without learning rate decay, on image size 256x256 pixels. The same number of iterations is used for training, i.e. 200 epochs on the subset day and night street view dataset meaning 380 000 iterations. This means the networks trained on the larger dataset only experiences each training image one time.

4.3.4 Other translations

To further investigate the capabilities and possibilities of the CycleGAN model, another domain translation is tested on 256x256 pixel images. The translation is between sunny and rainy street view conditions.
Results for all tests regarding both MR and street view images are presented in this chapter. For the MR evaluation, described in the previous chapter, both quantitative and qualitative results are shown. The quantitative results, and results from the perceptual study, are presented as bar plots to visualize the performance of the evaluated models. The street view images are only evaluated qualitatively. The images shown have been chosen carefully in order to have them represent the performance of the models in the evaluation.

5.1 Synthetic MR data results

The quantitative and qualitative results regarding the MR images are presented below.

5.1.1 Quantitative results

In the MAE evaluation the model that has the best result for both T1 and T2 images is the $G_s$ as illustrated in Figure 5.1a. The ratio of the error for the models between T1 images and T2 images is similar, with an increased error for T2 images. The model with the largest error is the Simple model. The PSNR for the different models is illustrated in Figure 5.1b. The model that has the highest PSNR is the $G_s$ model. The Simple model has the lowest PSNR. The MI for the different models is illustrated in Figure 5.1c. The $CG_{bl}$ and $VAE-C$ model have similar results. The model that performed best with the highest MI for both domains is the $G_s$. 
Results

(a) Mean absolute error

(b) Peak signal noise ratio
5.1 Synthetic MR data results

![Graph showing average mutual information for different methods and domains.]

**Figure 5.1:** Quantitative results from the MR evaluation. All results are averages over all 426 test images, 213 in each domain. The results for mean absolute error are shown in (a), PSRN results in (b) and Mutual information results in (c).

5.1.2 Qualitative results

For every evaluated method, 213 synthetic images in both the T1 and T2 domain were generated. Figure 5.2 shows samples from each method. The images on the most right-hand side and left-hand side are calculated as the absolute difference between the real image and each synthetic image, divided by the real image. Division by zero results in the value 100%. It can be seen in Figure 5.2 that the synthetic T2 images differ more over the grey and white matter than the synthetic T1 images does from the real T1 image. It can also be seen that the Simple and UNIT model has a high relative error for T2 images in the white matter and in the cerebrospinal fluid compared to the other models. The UNIT model has pattern artifacts in the image which is not visible in the synthetic images from the other models. To have in mind when examining the images in Figure 5.2 is that the generated image is highly dependent on the input, the quality of the output image can differ between different input images.
Figure 5.2: Synthetic images from all evaluated models. The real images shown at the top are inputs that the synthetic images are based on, the real T2 image is the input that generated the synthetic T1 images and vice versa. They are also a corresponding pair, i.e. the same slice from the same patient. This means the top images are ground truth data corresponding to the images below them. The colorbar belongs to the images in the left and right columns which are calculated as the relative absolute difference between the synthetic and ground truth image. T1 results are shown in the far left column and T2 results in the far right.
5.1 Synthetic MR data results

Perceptual study

The results of labeling the images in the perceptual study as real or synthetic are illustrated in Figure 5.5 and 5.6. The percentage illustrated in the figures refers to the amount that the expert labeled correctly. The total percentage for images that are labeled correctly for real and synthetic images is 58.3%. From Figure 5.5 it can be seen that fewer T2 images were labeled correctly than T1 images, reaching a value of 47.9% against 68.8%. For the labeling of the synthetic images for the different models, which is illustrated in Figure 5.6, it can be seen that the UNIT model has the highest combined T1 and T2 value of 91.7%. Which means that the UNIT model has the worst performance among the models. Both of the CycleGAN models, CG_bl and CG_s, produced T2 images that have a lower values than the VAE-C and UNIT model, contrary to the CycleGAN models the VAE-C and UNIT produce T1 images that had a lower result. The model that performed best for both T1 and T2 images is the VAE-C, with a combined value of 41.7%. Figure 5.3 shows two synthetic images from the two models, VAE-C and CG_bl, that have the most real classifications and that were classified wrongly in the perceptual study.

![Figure 5.3: Both of the synthetic images were included in the perceptual study and was wrongly classified by the expert. The real images were not included in the test.](image-url)
After the labeling was done, images from the different models were shown and the following comments about the different models’ performance of generating synthetic images were received:

- The \textit{CG\_bl} and \textit{CG\_s} had similar appearance where synthetic artifacts appear at the edge of the brain.

- The \textit{VAE-C} model also has a synthetic appearance at the edge of the brain but the noise in the images has a more realistic appearance than the results from other models.

- The T2 images are in general more difficult to classify since the images are darker which makes it harder to discover if high frequency details are missing and if the noise in the image has an unnatural appearance.

- The noise generated from the \textit{CG\_bl} model does not have a realistic appearance. An example is shown in Figure 5.4.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{real_vs_synthetic.png}
\caption{One of the comments that was received after studying the synthetic images from the different models was that the noise in the images generated from the \textit{CG\_bl} does not have a natural appearance.}
\end{figure}
5.1 Synthetic MR data results

Figure 5.5: Perceptual study results from the labeling of the real and synthetic images. The T2 images are considered to be more difficult to label correctly since they are darker.

Figure 5.6: Perceptual study results from the labeling of synthetic T1 and T2 images produced by the different models.
5.2 Synthetic streetview data results

The following section contains results on street view images. The first results are on the top and bottom five images from the CycleGAN and VAE-CoupledGAN models using settings according to the respective articles.

In Figure 5.7 the top five and bottom five synthetic images in both domains, when considering visual realness, from the CycleGAN baseline model on 256x256 pixel images, are shown.

**Figure 5.7: Synthetic results from the CycleGAN baseline model using the street view test dataset of image resolution 256x256 pixels.**

The top five synthetic CycleGAN images show realistic night results. A closer look on the day images reveals that many details are not visually realistic, but the images still have characteristics of day images. The five bottom images are poor in visual realness in both domains. Interesting mistakes are made, e.g. a road light is made into a traffic light and some pedestrians are turned into a crosswalk.
The results also show that the outputs are not consistent meaning the quality of the results differ a lot.

Results for the VAE-CoupledGAN model are shown in Figure 5.8.

**Figure 5.8:** Synthetic results from the VAE-CoupledGAN baseline model using the street view test dataset of image resolution 256x256 pixels.

The top results in Figure 5.8 are not as good as the results from the CycleGAN model. Road lines are kept and the overall visual appearance is as wanted, the problems are however that cars adapt the color of the road, making them invisible, and the details are not very sharp. The bottom results show that the model sometimes has trouble knowing where the road, trees and sky starts and ends.

The best performing model on image size 256x256 and streetview translations between day and night image domains is the CycleGAN model.
5.2.1 Results after identity training

The images in Figure 5.9 show different results for different amounts of identity mapping during training of the CycleGAN model. The results prove that increased amounts of identity training improves the results for translation of street view images of tunnels, both to the night and the day image domains. More identity training results in better details and overall color mapping. Translated images from tunnel to day have cracked ceilings where blue sky is created and this effect is decreased with more identity training. Even with identity training of $n = 100$, a clear difference appears compared to the CycleGAN baseline result.

The amount of identity training also affects the regular translations between the day and night domain. Figure 5.10 shows that more identity training results in less alternations to the input image. The characteristics of the target domains disappear with more identity training. The synthetic day image from the generator trained with $n = 5$ has the day characteristics of a blue sky and green trees. The differences it has from the images synthesized from the generators with less identity training however, is the car tail-lights and the street lights which are lit. When looking at the synthetic night images problems with coloring the sky black appear more and more with increased amount of identity training. Even with identity training of $n = 10$, some blue pixels close to a tree are kept blue.
5.2 Synthetic streetview data results

![Image](image_url)

**Figure 5.9:** Results of tunnels translated between the night and day domains from CycleGAN models trained with different amounts of identity training. All results are after 100 epochs of training. The same image is used as input for all generators since tunnels look the same during night and day.
Figure 5.10: Results of translations between day and night using the Cycle-GAN models trained with different amounts of identity training. All results are after 100 epochs of training.
5.2 Synthetic streetview data results

5.2.2 Higher resolution images

Figures 5.11 and 5.12 show results on images with 512x512 resolution where 5.12 are results where resize convolution has been applied on the CycleGAN model. Figure 5.12 also contains a result from the VAE-CoupledGAN model.

Figure 5.11: Results when training different CycleGAN models from scratch, 100 epochs on the small day and night street view dataset. Results in models with updates described in Section 3.2.2 are shown in 5.11c, 5.11d and 5.11e.
Figure 5.12: Results of the same training as done to generate the images in Figure 5.11. Results from the CycleGAN model updated with resize convolution are compared to the same training using the VAE-CoupledGAN model. The input image in Figure 5.11a was used.

The CycleGAN model does not provide good results without any modifications when the resolution is increased to 512x512 pixels. The multi-scale discriminator improves the overall night appearance and together with the expanded generator network the details are improved. All synthetic images in Figure 5.11 contain a checkerboard pattern which is reduces in Figure 5.12 where the resize convolution is introduced. The results with the resize convolution however loses some overall night characteristics. In terms of visual appearance, the best result on 512x512 pixel images is shown in Figure 5.12c and is produced by the VAE-CoupledGAN model. Issues with colors of the cars exist and the image is relatively dark, but the details are sharper.
5.2 Synthetic streetview data results

Figure 5.13: Test results during training of the CycleGAN model with baseline settings on 512x512 resolution.

(a) Epoch 55  (b) Epoch 57  (c) Epoch 58

Figure 5.13 shows that the checkerboard pattern appears and disappears during training.

5.2.3 Results on larger dataset

Figure 5.14 shows top and bottom five results for both directions of the translations between night and day street view image domains. In all translations to the night domain, the images have adapted colors common in city environments, independent of the actual environment. For example, they have lights in places like trees and road crash barriers. The translations to the day domain are of varied quality. The top five is generally good, but lacking details on cars. The bottom five is poor in quality and it is difficult to see what is represented in the images.

The results from the training on the large dataset is shown in Figure 5.15. The top five synthetic images in both image domains are more realistic than the ones produced by the model trained on the subset of the large dataset. The bottom five day images are also more close to realistic than the ones from the model trained on the subset.
Figure 5.14: Synthetic results from the CycleGAN baseline model on 256x256 pixel images. The model is trained using the subset dataset of the large day and night dataset. Top and bottom results are from city and open road street view environments.
Figure 5.15: Synthetic results from the CycleGAN baseline model on 256x256 pixel images. The model is trained using the large day and night dataset. Top and bottom results are from city and open road street view environments.
5.2.4 Other translations

Figure 5.16 shows results using the CycleGAN baseline model trained on the Veoneer street view dataset with sunny and rainy images on open road.

![Real sunny - Synthetic rainy](image1)

![Real rainy - Synthetic sunny](image2)

**Figure 5.16:** Results from translations between sunny and rainy street views. Input images are to the left and corresponding synthetic images are to the right. Relative to all test results, row one and three show visually good results, and row two and four show bad results.

The translations between sunny and rainy street view image domains show visually realistic results on 256x256 pixel images. It proves that good results are possible also in this particular translation, but it also shows an inconsistency since not all results are of the same quality.
The chapter contains the discussion of the results obtained in Chapter 5. The results on the MR images are first discussed and interesting data is pointed out and related to the theory part of the thesis. The results from the street view data are then discussed and similar to the MR data, interesting outcomes are related to the theory part of the thesis. In the last part of the chapter the applied method is discussed and criticized.

6.1 Results on MR images

In this section the results regarding the MR images are discussed.

6.1.1 Quantitative results

The results obtained from the MAE evaluation in Figure 5.1a showed that the $G_s$ performed best among the models. During training the $G_s$ model uses MAE as its only loss function, this creates a model where the goal is to minimize the MAE. The model does this well compared to other models which is shown in the result. The Simple model, which similar to the $G_s$ is only trained on the MAE loss, has the highest error among the models. The main difference between the Simple model and $G_s$ is the architecture. Where the Simple model only has two convolutional layers and the $G_s$ has, similar to the CycleGAN generators, 15 layers. This indicates that the architecture in the Simple model is not complex enough for the translation. The kernel size of the layers in the Simple model is one, which allows it to only train and perform a pixel intensity adjustment. It can be seen in Figure 5.2 that the Simple model lacks knowledge of the spatial translation where in the cerebral spinal fluid the translation between T1 and T2 images has a higher error than the $G_s$ model.
The \textit{CG\_s} model shows a slight improvement in MAE for T2 images compared to \textit{CG\_bl}. A possible explanation for the difference in MAE is that \textit{CG\_s} bases its loss on the MAE between the synthesized image and the corresponding ground truth images. Which means that during training the \textit{CG\_s} learns to minimize the MAE. Interesting results are that even though the ground truth images are available, the results are not significantly better than \textit{CG\_bl}, which has been trained unsupervised. In fact, the MAE on T1 images is better for \textit{CG\_bl}. The \textit{CG\_bl}, \textit{VAE\_C} and \textit{UNIT} show similar results and it is difficult to argue why one or the other performs slightly better than the other one. The synthetic T2 images differ more in MAE, which can be seen in Figure 5.2. This indicates that the generation of T2 images seems to be more complex than the generation of T1 images.

The PSNR results shown in Figure 5.1b show similar results as the MAE for the different models in Figure 5.1a. This because the PSNR is calculated with the MSE which is closely related to MAE. The main difference is that outliers get a higher error in the MSE than in the MAE measurement, as explained in Section 2.2. Another difference between the MAE and PSNR is that the T1 and T2 images have more equal values in the PSNR measurement, this is partly because of the logarithmic function that is used in the calculation of PSNR.

From Figure 5.1c it can be seen that T1 images have a higher value for MI than T2. This can be correlated to the results from MAE where a larger error was generated from the T2 images. It can also be seen that the \textit{Simple} model has a lower score compared to \textit{G\_s} which indicates that the complexity of the model has an effect on the performance. The architecture of the \textit{Simple} model is not complex enough.

An explanation for why the \textit{Simple} model performs better than the majority of models for T2 images is because T1 and T2 images from the same patient contain a lot of similar information. Since the \textit{Simple} model only change the intensity in the pixels, a lot of information is preserved in the transformation between T1 and T2. There is also a large difference between the \textit{UNIT} and \textit{VAE\_C} which indicates that the models differ from each other. The differences in the data preprocessing for the models could be a factor that has a large effect. The \textit{UNIT} model also uses a data augmentation method, mentioned in Section 4.2. Differences in the deep learning frameworks might also have an effect.

\subsection{Qualitative results}

From the qualitative visualization in Figure 5.2 it can be seen how the different models perform when translating the images into the different domains. For both the T1 and T2 images there is a high error on the edge of the brain which can be explained by that each individual has a unique form, the area is not homogeneous. The areas where there is an intensity change, e.g. CSF and white matter, seem to be more difficult for the models to learn, this might also be due to the individual differences between patients.

The relative absolute difference of the background is higher on the T2 results from the \textit{UNIT} and \textit{Simple} models which makes it dark red. This is due to that the mean intensity is subtracted during normalization before generating the rel-
Results on MR Images

The subtraction of the mean intensity effects all pixel values in the image and a larger difference gives a higher value in the background.

From the perceptual study it was shown that the synthetic images have a visually realistic appearance since synthetic images where classified as real images. It was also shown that T2 images are more difficult to classify than the T1 images. The reason for the difference could depend on that the synthetic T2 images had a more realistic appearance but also due to the fact that the expert has had different experience from the two image domains. If comparing the different models it can be seen that the synthetic images from the UNIT model was the least visually realistic. From Figure 5.2 it is also shown that the UNIT model has the most obvious synthetic pattern among the different models, and is probably the explanation as to why they were easy to classify as synthetic.

The VAE-C and UNIT model differ a lot when comparing the synthetic images from the models, and it is not clear as to why. The fact that the UNIT model uses a data augmentation flip during training should intuitively help the model to perform better, but in the perceptual evaluation the result is the opposite.

It is shown by comparing the two different models, CG_bl and CG_s, that supervised training does not generate images that are more realistic than training unsupervised. This indicates that the loss, which is calculated by the MAE between the synthetic images and corresponding ground truth images, in the CG_s, gives the images a more unnatural appearance. A theory of what is causing the less realistic appearance is that the generated images have a natural appearance, but that appearance is not similar to the ground truth image. The CG_s penalizes the appearance that is not similar to the ground truth, since it uses the MAE loss during training, which forces it to another direction, closer to the smooth appearance of the images from the G_s model. If the aim of the test would instead be to evaluate how similar the synthetic images are to the ground truth, the translated images from CG_s might give a better result.

By comparing the results from the perceptual study and the quantitative evaluation, no obvious correlation can be found. The VAE-C model had the best score for T1 images in the perceptual study, but had an average score in the MAE and PSNR evaluation. CG_bl had the best score in the perceptual study on T2 images, but had the lowest score in the PSNR evaluation, if excluding the Simple model.

If the aim was to create images that are as similar to ground truth images as possible, the quantitative measurements would be more applicable. This because the quantitative measurements are calculated using ground truth images. What is clear is that even if a model such as Simple has a relatively good score in the quantitative measurements, it is not necessarily visually realistic. This indicates that solely determining if an image is visually realistic can not be done with the used metrics.
6.2 Results on street view images

In this section the results regarding the street view images are discussed.

6.2.1 Identity training

The identity training implementation used in this thesis differs from the original CycleGAN implementation in PyTorch, where an identity loss weight controls the amount of identity training. Both implementations were tested to make sure similar results were attained. The difference in the results from the two identity training methods was that the method with a loss weight seemed to be more stable during training, meaning results on test images differed less between epochs. However, similar results from the two implementations were attained after enough training, about 100 epochs. The advantage of the method with identity training every n:th iteration is that the training time is shorter. This is because the weights are not updated each iteration.

The aim of the identity training was to handle the special case of tunnels in the street view datasets. Figure 5.9 shows results where the aim is achieved. This result is not surprising since street view images of tunnels look very similar to street view night images. The identity training teaches the network to preserve images that already have the characteristics of the target domain. The tunnel image translated to the night domain does not need a lot of modifications to obtain the appearance of a night image. The $CG_{bl}$ model alters the tunnel image more than the other models and the translated image contains less details as a result. More interesting are the results of the tunnel image translated to the day domain. The tunnel image does not have the characteristics of a day image, still the identity training has an affect of the wanted ceiling preservation. The difference between the output from the CycleGAN baseline generator and the generator trained with identity training with $n = 100$ is apparent. The problem with the ceiling is still there, but the breach is smaller, and the details on the cars, road and walls are better. Details generally improve with more training, and since added identity training means additional training iterations, some differences might depend on that.

Identity training obviously can help with some special cases, but it should not be forgotten that it has an effect on the performance of the generators intended translation capabilities. Figure 5.10 show translations between night and day using the same models used to produce the images in Figure 5.9. It can clearly be stated, by looking at the synthetic day images, that the output from the generators trained with identity training of $n = 2$ and $n = 1$ does not perform as wanted. Differences between the other images are also visible. The results comes from the fact that while the generators are trained to keep certain characteristics of some images, too much identity training makes it hard to understand which characteristics are unique to the target domain. Instead of converging towards a true translation function, the generators outputs images that are too similar to the input images, independent of the input.
By looking at Figures 5.9 and 5.10, \( n = 10 \) might seem like the logical choice, but more samples have to be evaluated before such a decision can be made. This because there can be big differences between samples. A trade-off between special case handling and its effects on the intended translation must be done in this situation, and similar situations.

### 6.2.2 Higher resolution images

Before jumping to conclusions regarding the images in Figures 5.11 and 5.12 it should be noted that different results are attained during different iterations of the training. First of all, the obvious patterns in the result images from the CycleGAN baseline model on 512x512 pixel images, with and without expanded generator architecture, was not present throughout the training. Figure 5.13 shows that the pattern appears and disappears during training. The training for both models was however unstable and even though the generators during some epochs were moving their outputs towards the respective target domain, they ended up with the poor result that is presented. Since the image size has been increased it is very likely that hyperparameter adjustments are necessary, and could lead to better results. Architectural modifications might also be needed for better details on higher resolutions.

Without any hyperparameter modifications, the model update of the multi-scale discriminator seemed to improve stability during training whilst also improve the results. Images 5.11d and 5.11e are clear improvements compared to images 5.11b and 5.11c. Still there are two issues; sharp details and checkerboard patterns. An attempt to handle the checkerboard pattern was implemented by the resize convolution and the method worked surprisingly well. Images 5.12a and 5.12b does not contain any checkerboard pattern but it is obvious that the modification also affects the learning of the domain translation. The resize convolution applied on the CycleGAN baseline model made the training converge to the presented result quite quickly. When applied on the model with both the expanded generator architecture and the multi-scale discriminator, the training was stable. A problem with the resize convolution could be that it blurs the images a bit, which can be seen by looking at the cycled images during training, compared to without the resize convolution. Since the CycleGAN model depends a lot on the cycle-consistency loss, this might be the reason for the performance. Even though the final results was not satisfactory, the resize convolution handled the checkerboard artifact well enough to be considered an improvement. In this case it is also likely that hyperparameter optimization could result in improved synthetic images.

A very interesting results is found when comparing the synthetic results to the image generated from the VAE-CoupledGAN model with its baseline settings. Despite improvements to the CycleGAN model, the results are not as visually realistic as they are on 256x256 resolution. Results from the VAE-CoupledGAN baseline model are however equally realistic on both 256x256 and 512x512 resolution. The image in 5.12c has sharper details than all other synthetic images in Figures 5.11 and 5.12 but has the issue of cars that adapts the color of the
Another issue is that the image is relatively dark. Which result is the best might depend on the eye of the beholder but it should be safe to say that checkerboard patterns are not present in a visually realistic result. The two candidates of best results are the images in 5.12b and 5.12c where the results from the VAE-CoupledGAN model can be considered better if sharper details are preferred. However, the synthetic images in 5.13a and 5.13c are considered as more visually realistic than the VAE-CoupledGAN results, despite the unstable training.

6.2.3 Effects of a larger dataset

The results in Figure 5.14 and 5.15 show that the increased size of the dataset can have a positive effect. The top five images are of higher quality. Both models generate synthetic night images that contain a lot of light. The model trained on the larger dataset however places the lights in more reasonable positions, and avoid lighting up objects like trees.

The vast improvement using the larger dataset is in the presented results obvious. The lacking result from the model trained on the subset can however be due to the training images. Since these have been selected randomly from the larger dataset images, the distribution of street view scenarios and other factors in the images, might not be similar to the distribution of the larger dataset. The results might have differed if another subset of images was created and used to train a new model. More tests need to be done before conclusions can be drawn with confidence, but the results definitely hint that an increased dataset size is positive when using these models.

The results in Figure 5.14 and 5.15 can also be compared to the results from the CycleGAN model trained on the small day and night street view dataset which can be seen in Figure 5.7. All top five images, from both models, are about as visually realistic. The key difference is that the model trained on the larger dataset can handle input images in other environments than city better than the model trained on the small dataset. The output images are also more diverse. For example, an interesting feature that the model trained on the larger dataset learns, is to output images with both blue and cloudy sky. However, the model trained on the small dataset still generally handles city images better. Only images captured in a city environment were allowed in the small dataset which simplifies the translation task. It is reasonable to believe that a simpler translation task needs less training data, which might be the reason for the visually realistic results of the top five night images in Figure 5.7. The bottom five however, contain some artifacts that might be the results of overtraining. These images could in that case have been improved with a larger dataset size.

6.2.4 Other translations

The result of the translation between sunny and rainy street view image domains in Figure 5.16 again proves the adaptive quality of the CycleGAN models to new translation tasks. The most visually attractive results shown in row one and three in the figure have great detail. Road lines, cars and even trees have sharp and
realistic appearances. In the examples in row two and four however, the details are not as good. In fact, generally the translations of objects were worse in the translations between sunny and rainy compared to translations between night and day. It is likely that this depends on differences in the datasets where the city environment of the day and night datasets contains a lot more objects like cars and road signs. Especially for cars, the amount of instances in the datasets appears to be reflected, to some extent, in the result.

An interesting thing to note regarding the translation between the sunny and rainy street view image domains is that most of the samples in the dataset with rainy images appear to be blurry because of the rain. The rain acts as a noise and blurs out edges and other details in the images. The generator is then forced to try and create detail from noise when translating these types of images to the sunny domain. This demand resembles the task that the night-to-day generator tries to achieve when creating objects like trees and buildings from groups of basically completely black pixels. As seen in Figure 5.16 when comparing the translations from rainy to sunny, the input image containing a lot of rain similar to noise results in an output image with far worse details. This can easily be seen by for example comparing the trees in the synthetic image on row three, to the car of the synthetic image on row four.

6.3 Method

The choice of implementing and testing two GAN models was made early in the work and it was shown in Chapter 5 that the chosen models synthesized realistic images. An alternative method would be to implement more models and compare them against each other. The other models could possibly synthesize images with a more visually realistic appearance than the implemented models. The drawback would be that fewer versions of the CycleGAN model would have been implemented due to time limitation. The implemented variants of the CycleGAN model created a better understanding of GANs in general, which helped when optimizing the model for synthesizing higher resolution street view images.

When investigating the models a delimitation was to not optimize the hyperparameters and instead focus on architectural changes and model modifications. If for example hyperparameter grid searches would have been included for the different models better results might have been obtained. Because of the time limitation for the work it was chosen to exclude hyperparameter searches and rely on the suggested settings from the original implementations.

The results from the MR evaluation gave interesting results when the UNIT model was outperformed by the other models in the perceptual study. The method for preprocessing the data before training the different models is however different for the UNIT model. A fairer comparison would have been to process the images in the same manner for all models.

What was discovered when training a model was that the images could for a series
of epochs contain artifacts and then in later epochs appear without artifacts. The models used for the MR evaluation was selected by visually inspecting the images produced by the models during training. A possible improvement would be to perform the quantitative evaluation for a model trained for a different number of epochs. The best performing model would then be used in the comparison between the different models.

The perceptual study is the most valuable result to answer the research question on how visually realistic results can be obtained with the chosen models and datasets. The study that was made was small, both in the number of test images and test participants. Many more participants and image samples from each model could have provided a statistically trustworthy result where a fair comparison between the models could have been done.
The purpose of the thesis was to investigate the possibilities of image-to-image translation using GANs on medical MR and street view images. Among the investigation was the goal to generate as visually realistic images as possible by comparing and implementing different models. In order to assess the visual appearance of the generated images, and to compare the different models, methodologies for evaluation had to be derived.

Synthetic medical MR images have been generated using different models. Both quantitative and qualitative results have been provided via pixel wise error measurements and a perceptual study. On street view images, investigations on different image resolutions, training dataset sizes, translation tasks and GAN models have been made. Results have been compared by visual inspections. Also, modifications to proposed models have been made in order to expand the exploration. The investigations have resulted in answers to the stated research questions addresses in the following section.

7.1 Answers to the research questions

Can GANs be used to generate synthetic images from data provided by Veoneer?
Synthetic images can be generated. Both via translations between day and night, and sunny and rainy image domains.

Can GANs be used to translate between T1 and T2 weighted MR image domains?
Translations between T1 and T2 MR image domains have been shown using different GAN models.

How visually realistic can the synthetic images become?
Synthetic MR images with visually realistic appearance comparable to real MR images have been generated. This has been proven via a perceptual study. Varying results are however attained and some images can contain artifacts even from models with generally realistic output images.

Regarding street view images the image-to-image translation task is more complex compared to the MR images. Since the concept of realistic appearance in images is subjective, it is difficult to specify how visually realistic results have been attained. Our opinion is that realistic synthetic images have been generated, but the results vary between the image domains. Also, the average quality of the synthetic images is not very realistic and the how visually realistic the output is differs a lot between input samples.

*How does the implemented GAN model perform when results are compared to ground truth data?*

Comparison to ground truth data has been made on the synthetic MR images. A perceptual study has shown that the synthetic images from different models have visually realistic appearance comparable to real images. Quantitative measurements show results from the GAN models are not far from the results from a simple neural network model. Together with the perceptual study result, this means that the quantitative measurements can not be relied on to evaluate how visually realistic synthetic images are.

### 7.2 Implications

Understanding of the opportunities with GANs, in the synthetization of visually realistic images, provides a platform for development of models for more complex data augmentation methods than the ones common today. This could prove to be valuable in the automotive vision industry and the medical imaging field, but also any other field or application area where synthetized data is of interest. From the exploration made in this thesis, information that can be used as a base for further development is provided.

### 7.3 Future work

A suggestion to future work is to investigate if GANs can be used as a data augmentation method. A possible way to answer the question would be to test it on a classification network for semantic segmentation. GANs would be used to generate synthetic training data for the classification network. Several test could then be done with different amount of synthetic data. The performance of the classification network trained with different amount of synthetic data could then be evaluated. Which would indicate if GANs can be used as a data augmentation method.

When training the different GAN models it has been discovered in some cases that the training of the networks is unstable. The generator could for example
generate images with good detail quality without artifacts, and then later in the training generate images with less detail quality and artifacts. This indicates that the hyperparameters are not optimized for the training. An optimization could potentially give a more stable training.

Another suggestion for future work is to continue the investigation of how the increased dataset size effect the generation of images. In this thesis only a few tests were performed with larger dataset where a clear improvement could be seen on the synthetic images. More tests are needed to draw a conclusion. Suggestions of such tests would be to train more subsets of the larger dataset. This to strengthen the results that a larger dataset is improving the performance of the network. It would also be interesting to perform longer training on the dataset to investigate when the training starts to converge. Future work could also be performed on images with higher resolutions to investigate how large images the networks can handle. This in combination with larger dataset could potentially generate interesting results.

Since MR data is represented as 3D volumes the generation of individual 2D images is not sufficient for a translation of a whole human brain. The 3D information connecting the images from each slice of the brain is missing in the generation of the synthetic images in this thesis. To handle 3D volumes the implementations in this thesis would have to be modified. Since the MR images are grayscale and thereby only have one image channel, the input volume can be expressed in three dimensions and the modifications to the code is therefore intuitively simple. However, an MR volume can contain over 200 image slices which requires that the architectures of the GAN networks are magnified. The magnification will result in an increase of the number of parameters to be trained and the needed memory usage will also increase. For example, the added encoding and decoding steps in the expanded generator architecture of the CycleGAN model increases the number of parameter of the network from approximately 2.5 million to 10 million parameters. The needed increase to get a good result on 3D volumes is likely to be a lot larger.

Generating 3D volumes with GANs has been showed possible and one example is the work by Wu et al. [35]. The results are however not as complex and detailed as an MR brain volume and future work is needed to acquire a visually realistic result.
Bibliography


