Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Datateknik
2021 | LIU-IDA/LITH-EX-A--2021/043--SE

Exploring the Capabilities of Generative Adversarial Networks in Remote Sensing Applications

David Nyberg

Supervisor : George Osipov
Examiner : Cyrille Berger

LINKÖPING UNIVERSITY

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

The field of remote sensing uses imagery captured from satellites, aircrafts, and UAVs in order to observe and analyze the Earth. Many remote sensing applications that are used today employ deep learning models that require large amounts of data or specific types of data. The lack of data can hinder model performance. A generative adversarial network (GAN) is a deep learning model that can generate synthetic data and can be used as a method for data augmentation to increase performance of data reliant deep learning models. GANs are also capable of image-to-image translation such as transforming a satellite image containing cloud coverage into one without clouds. These possibilities have led to many new and exciting GAN applications.

This thesis explores ways generative adversarial networks (GANs) can be applied in a variety of remote sensing applications. To evaluate this, four experiments using GANs are implemented. The tasks are: generating synthetic aerial forestry imagery, translating a satellite segmentation mask into a real satellite image, removal of thin cloud cover from a satellite image, and super resolution to increase the resolution of a satellite image. In all experiments the tasks were deemed successful and prove the potential for further use of GANs in the field of remote sensing.

Keywords: Generative Adversarial Networks, Data Generation, Remote Sensing

Acknowledgments

I would like to thank the company Arboair for much of the motivation and discussion of the ideas behind this thesis as well as the opportunity to use their hardware and data in parts of the work. Arboair is a startup company focusing on sustainable forestry and detection of forest damage using machine learning models.

I also want to thank my supervisor George Osipov who gave me feedback throughout the process of writing my thesis and my examiner Cyrille Berger.

Contents

List of Figures

1 Introduction

This chapter provides an introduction and motivation to the thesis, the overall aim, the research questions to be answered, and the delimitations to be considered.

1.1 Motivation

In recent years the rapid growth of deep learning has made it a popular technique to analyze remote sensing data. Deep learning approaches in computer vision use large models that consist of artificial neural networks with several layers, also referred to as deep neural networks. These networks are capable of learning to understand the context in images for tasks like object detection and image classification. Applying deep neural networks within the field of geospatial analysis using remotely sensed imagery includes tasks such as land use and land cover classification, semantic segmentation, object detection, and change detection [31]. Traditionally, these tasks have been accomplished with machine learning techniques such as random forests [3], support vector machines [33], or other pixel-based approaches. While these aforementioned methods can achieve desirable results, the advances in deep learning models for computer vision tasks have begun outperforming previous methods [31].

A major event in the improvement of computer vision was the introduction of AlexNet [24] in 2012. Alex Krizhevsky and his colleagues introduced their network to compete in the ImageNet competition [38] which is a challenge where researchers compete to achieve the best results in object detection and image classification on 1,000 classes in over 1.2 million images. Their convolutional neural network (CNN) performed over 10% better than the second place model. Apart from their larger network architecture, they explain the necessity of large training datasets and how the advancements of graphical processing units (GPUs) allow for optimized convolutional operations.

Since the majority of remote sensing data is imagery that CNNs perform well on, the jump to use deep learning with remotely sensed imagery is an apparent one. However, requiring thousands or even millions of images to train a model is an issue that many deep learning approaches have. Within remote sensing, the data acquisition can be expensive or time-consuming making it difficult to gather large enough datasets for training these models. Methods have been introduced to help create more data for computer vision tasks such as data augmentation [40]. Data augmentation is the process of altering the already existing data with techniques like changing color profiles slightly or rotating and cropping the image. Using augmentation techniques will introduce additional training samples and diversify the data which can help train large models by improving its generalization capabilities.

The quantity of data is important, but the diversity of data is equally relevant for the performance of deep learning models. With an uneven data distribution, it will be difficult for a model to learn how to correctly detect or appreciate all the features in a dataset. For example, remotely sensed UAV (e.g., drone, aircraft) imagery of forests can be used to detect changes in forest health by performing classification and detection at an individual tree level [35]. In the case for forest health, the data collected might result in a distribution of less than 10% of dead or sick trees, whereas the majority would be healthy trees. Data augmentation could help fix this unevenness by augmenting only images containing the less represented class and therefore increasing their contribution. In some situations, introducing these images might lead to sufficient results. However, getting an equal distribution of data is still difficult because there is a limited number of ways to meaningfully augment the existing images. Therefore, a technique to generate new synthetic images could be used. One such method explored in this thesis is the generative adversarial network.

The generative adversarial network (GAN) [11] is a deep learning model that has the ability to generate synthetic data. When given examples of images, GANs are capable of learning the underlying data distribution which allows for generation of new imagery that does not exist in the training set but contains the same characteristics as images that do. By using this learned distribution, a trained GAN can continue to generate as much data as needed. This allows the usage of new synthetic samples as a technique for fixing the unevenness that may be present in a dataset or simply increasing the dataset size altogether. In addition to generation, the GAN architecture can perform image-to-image translation [15] where the characteristics of one image is transferred onto the other. This is also useful as a controllable augmentation technique where the type of data can be specified by the two domains that are being transferred between. In the field of remote sensing GANs have been used in several different ways: image-to-image translation to transfer styles from basemaps to real satellite imagery [46], removing thin cloud cover from satellite imagery [42], and as a tool for super resolution to increase the resolution of remotely sensed images [36].

1.2 Aim

The aim of this thesis is to explore the capabilities and applications of generative adversarial networks in the domain of remote sensing. Specifically, the focus of this thesis includes: GANs as a tool for generating synthetic remote sensing data, image-to-image style transfer, removal of thin cloud cover from satellite imagery, and image super resolution. The data used includes satellite imagery and unmanned aerial vehicle (UAV) imagery.

A broad overview of some ways to utilize GANs in the field of remote sensing is given by evaluating different GAN architectures for the aforementioned tasks. The already existing usage of machine learning in remote sensing applications is a key factor why the introduction of GANs is important. For example, how the introduction of GANs can aid in data-centric machine learning models and how GANs can be used for better solutions to already existing techniques.

1.3 Research questions

This thesis focuses to explore and evaluate the following research questions:

1. At what resolution and quality can synthetic aerial forest imagery be generated with GANs?
2. Can a GAN learn the mapping between segmented satellite masks and real satellite imagery using paired image-to-image translation?
3. Can a GAN learn to remove thin cloud cover from satellite imagery using unpaired image-to-image translation techniques?
4. Can a GAN be used in a super resolution task using satellite imagery to generate higher resolution satellite imagery?

All of the research questions will be answered with individual experiments and evaluation is done mainly through a qualitative analysis using quantitative metrics to help determine the best performing models. The full evaluation techniques are described in the method chapter.

1.4 Delimitations

In this thesis, the application area where generative adversarial networks are evaluated is within remotely sensed imagery. The experiments conducted use imagery captured from drones and satellites and the imagery is all within the visible light spectrum (RGB). Only the aspects related to the GANs performance and quality of generation are explored. The applications of the generated data will be discussed but not evaluated further.

2 Theory

2.1 Background

This section will describe the important concepts needed to understand the work done in this thesis. First, an overview of remote sensing is given, then an introduction to generative adversarial networks is presented. For a complete introduction to deep learning refer to the Deep Learning Book [10] or for a condensed introduction refer to this [26] Deep Learning article.

Remote Sensing

Remote sensing can be broadly defined as the process of gathering information at a distance. More specifically, capturing the reflected or emitted electromagnetic radiation from the Earth's surface from an overhead perspective [5]. Remote sensing is a vital component for geospatial monitoring and analysis of the Earth. Remotely sensed imagery can be acquired by satellites, aircraft, or unmanned aerial vehicles (UAVs) and can contain data from multiple sensors to capture light outside of the visible spectrum. The resolution of remotely sensed imagery is often referred to as the ground sampling distance (GSD). The GSD describes the real distance one pixel in a remotely sensed image represents on the Earth's surface. Satellite imagery might be preferred if the area of interest is very large, whereas a UAV would be more useful in acquiring high-precision data over a small area. Imagery in the visible light spectrum might be most relevant for mapping urban cityscapes, whereas infrared light can be used in agriculture to measure crop health using a vegetation index such as the normalized difference vegetation index (NDVI).

Generative vs Discriminative

Generative and discriminative models are two distinct approaches to machine learning. Discriminative models learn to find differences in data. This can be done by optimizing a separation of the data with a decision boundary in order to create distinct groupings of the data. The boundary is then used as a threshold where any data above the threshold is of one class and any data located below is of another. Instead of just trying to distinguish the differences in data, generative models learn about the underlying distribution of the data.

Discriminative models find a direct solution by modeling the posterior $P(y|x)$, by predicting class labels y given input x. Generative models use Bayes rule to model the joint probability $P(y, x)$ to choose the most likely class label y [18]. Instead of directly calculating which class label a data point x should have, generative models compute an intermediate step $P(x|y)$ in order to model the distribution of a certain class label.

In more recent years deep generative modeling has gained much attention. These models learn the distributions of the data they are trained on and then use this distribution to generate new synthetic data. Many of these models can learn this distribution completely unsupervised. The advantage of not needing to obtain labeled data can be a tremendous time and money saver and is one reason why generative models have gained attention. One model that is commonly used in generation tasks is an auto-encoder.

A variational auto-encoder is a neural network architecture that can be used to generate synthetic data. It is a combination of two neural networks that consists of an encoder network that transforms the input into a smaller dimension space called the latent space, and a decoder network that effectively undoes the process of the encoder by using the latent space in order to reconstruct the input. The objective of this architecture is to minimize the error between the original input data and the reconstructed data. During training the auto-encoder learns the properties of the data distribution in a lower-dimensional representation that can later be used as the basis for generation. This is done by first removing the encoder network, then the latent space and the decoder can be used to generate data by taking a random sample in the latent space, and sending it through the decoder network [25]. Because the latent space will define the underlying distribution of the data the auto-encoder has been trained on, it allows the decoder to produce data that resembles the inputted training data.

The generative capabilities of variational auto-encoders is limited to how much the data can be compressed by the encoder and stored in the restricted size of the latent space. This is a problem that limits the ability to learn complex image representations and something that the generative adversarial network can solve.

2.2 Generative Adversarial Networks

The timeline of generative adversarial networks is relatively short, but many improvements to the original architecture have been made. This next section introduces generative adversarial networks, the important details regarding training, common issues, extensions to the GAN architecture, and the applications of GANs.

The Original GAN

In 2014 a novel architecture with the goal of generating new data was proposed by Ian Goodfellow et al [11]. The proposed generative adversarial network consists of two separate neural networks, a generator network and a discriminator network both of which are multi-layered feed-forward neural networks. The GAN architecture works by having these two networks train as adversaries, meaning they are competing against each other. The generator network simply tries to generate an output that resembles the training data starting from a random noise vector as input. The discriminator network then decides if any particular sample is from the training set (real) or created by generator network (fake). In the context of image synthesis, the untrained generator will initially generate an image of random noise. Consequently, the discriminator will look at these obviously fake images and classify them as such. The feedback that the discriminator could detect this image as a fake is given back to the generator in order for it to improve its generation. As the discriminator is also improving its ability to recognize real images, its ability to recognize the fake images improves as well [25]. As the generator network continuously improves its generation capability it will eventually begin to generate images that look exactly alike the images from the training set. At this point the discriminator will no longer be able to classify if the image is fake or real and reach the end of training.

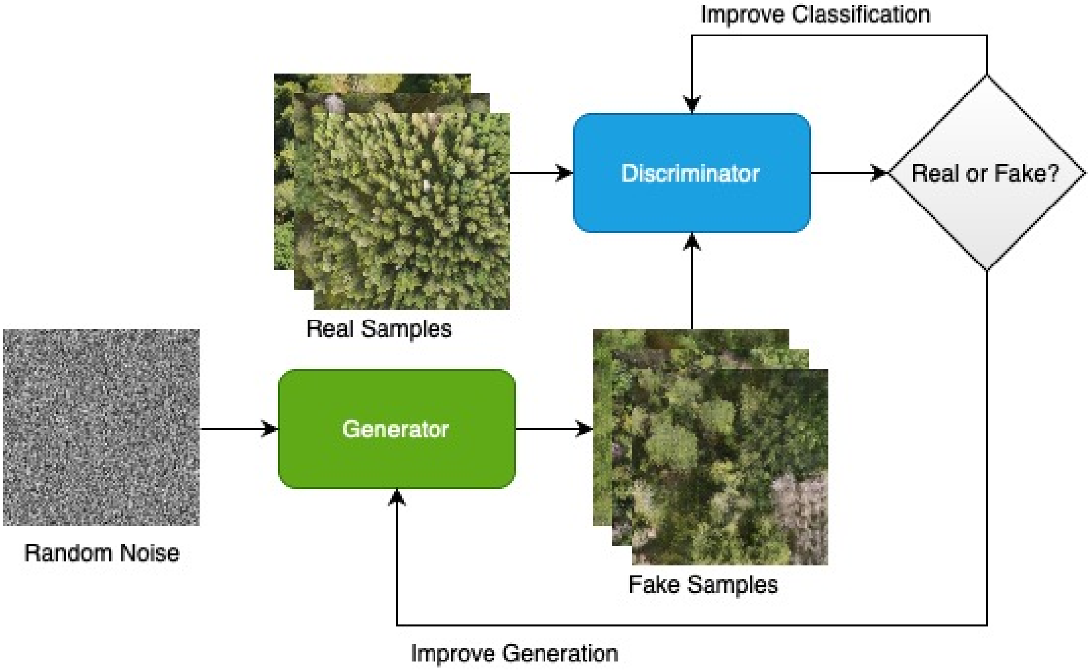

Figure 2.1: Simplified architecture of a GAN

Specifically, the training process for the GAN is a back and forth process where the discriminator learns to maximize its classification accuracy in order to detect if a sample image is real or fake. This is done by sampling a batch of real images and a batch of fake images both which are labeled accordingly. Since the labels are known the discriminator learns in a supervised manner in order to improve. Adversely, the generator tries to minimize the discriminator's ability to classify the generated image as a fake image. These two networks alternate to updating their weights through backpropagation during training to train evenly and reduce overfitting in one the networks. Due to consisting of two separate neural networks the GAN loss function is actually two separate loss functions. These two separate equations can be combined into what is described as minimax loss [11] expressed below as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]. \tag{2.1}$$

where D and G represent the discriminator and generator network respectively, and z is the random noise vector fed as input into the generator. D maximizes the classification accuracy over all samples. G only influences $1 - D(G(z))$, the probability that D detects a fake sample, which is to be minimized by G.

To further simplify and understand how GANs work Goodfellow [11] wrote a great analogy in his original paper, "The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles."

The results seen from the original generative adversarial network include generating numbers from the MNIST dataset [27], generating faces from Toronto Face Database [43], and generating many diverse scenes from the CIFAR-10 dataset [23]. While the early results of the first GAN model are impressive, Goodfellow et al. [11] leave many suggestions for extending and improving the generative adversarial network.

Difficulties with GANs

With traditional neural networks, it is straightforward to monitor the training and validation losses and then stop the training when they begin to diverge in order to avoid overfitting [25]. With GANs, however, this is not the case because two separate loss functions are being optimized each with a different goals. They do not converge like a traditional loss function, but is instead referred to as a minimax two-player game by Goodfellow [11]. A minimax, or a zero-sum game, is a competition between two players (the discriminator and generator networks in the case of GANs) and the solution to such a game is known as the Nash Equilibrium [34]. Proposed in 1951, Nash Equilibrium is defined as a point where neither competitor in a zero-sum game can improve their standing without changing their tactic. GANs reach this point when the generator starts generating images that are nearly identical to the training data and the discriminator has reached a point where it is uncertain if any data sample is real or fake. At this point guessing randomly if an image real or fake is the optimal choice for the discriminator [25].

A weakness of the original GAN discriminator is that it makes a simple binary decision if an image is real or fake. Proposed in 2017 by Martin Arjovsky et al [2], Wasserstein GAN (WGAN) provides methods that for the first time show properties of convergence of the loss functions in GANs. The authors introduce Wasserstein loss as a solution that changes the discriminator's binary classifier from the original GAN into a critic that produces a numerical value that describes how real or fake an image is [29]. The loss is based on the Earth Mover's distance which is the distance between the distribution of real images and fake images. The Wasserstein loss is represented as:

$$\min_{G} \max_{C} \mathbb{E}_{x \sim p_{data}(x)}[(c(x))] - \mathbb{E}_{z \sim p_z(z)}[(c(g(z)))]. \tag{2.2}$$

where c is the discriminator network which is now a critic, g is the generator and z is the input noise vector. This is similar to the minimax loss but with the removal of logarithms as the outputs from the critic are not bounded by [0,1] and the distance between two distributions is being calculated.

Another solution to improve the discriminator is the relativistic GAN [17] (RGAN). RGAN is a GAN implementation where the discriminator uses prior knowledge that half the samples seen are actually fake. Meaning that at convergence the discriminator should assign higher probability to a given sample being fake rather than being fooled that all the samples are real. Essentially, as the generator has begun to fool the discriminator, the discriminator should know the probability of a realistic image being real is actually lower than it was earlier in the training phase.

A common problem that can occur when working with GANs is mode collapse. This occurs during the training phase when the generator network gets stuck producing the same set of images despite the training data containing other samples [25]. In practice, this can be detected when the images being generated are all similar in appearance. In the case of heavy mode collapse, all generations could be the exact same image. The original GAN discriminator evaluates data samples one by one which easily leads to mode collapse due to never comparing the generated images to one another. This makes it impossible for the network to detect if the generated images are similar. Therefore, by always receiving good feedback by producing any image that fools the discriminator, the generator can continue producing the same image [10].

One solution to help mitigate mode collapse is minibatch discrimination [39] which allows the discriminator to evaluate a whole batch of generated images instead of just one at a time. This is similar to batch normalization [14] which shows how normalizing every batch of training samples which can reduce training times by allowing for higher learning rates. However, in minibatch discrimination, instead of normalizing the whole batch of inputs, the similarity between the batch of images is calculated and then used as extra information to help the discriminator decide if a generated image is real or fake. With the information about the similarity between images, the discriminator will learn to penalize the generation of similar images, therefore, steering the generator away from mode collapse.

Evaluation of GANs

In addition to all the aforementioned difficulties that can occur when training generative adversarial networks, yet another issue arises when evaluating them. Due to the adversarial training procedure, it is possible for a GAN to never converge. This makes evaluating the loss an unreliable metric. Because of this fact, several evaluation metrics have been introduced but no standard method for evaluation has been decided upon and differs between researchers. This has led to a combination qualitative and quantitative assessment of the quality of generation being the best way to assess the performance of a GAN [39] [4].

One metric that is used to assess the quantitative performance is the Fréchet Inception Distance (FID) [12]. FID is a metric that can be used to evaluate how similar generated images are compared to the real training images. FID is calculated by using the last feature embedding layer from a pre-trained Inception V3 model (a large CNN used for object detection) is used to compare the distributions of fake and real images. By inputting samples of generated and real images into this feature layer, two respective feature vectors are obtained. These two vectors are modeled as multi-dimensional gaussian distributions where the distance between the two is measured by the Fréchet distance. The FID is calculated as:

$$FID = \|\mu_g - \mu_r\|^2 + Tr(\Sigma_g + \Sigma_r - 2\sqrt{\Sigma_g \Sigma_r}). \tag{2.3}$$

where r represents the real images, and g are the generated images. The distance is calculated as the difference between the means μ plus the variation of the respective distributions. By taking the Trace (Tr) of the matrix representing the covariance Σ for each distribution, the variance is calculated. The closer the two distributions are the lower distance between them, meaning that the real and fake images are similar. Since this metric relies on modeling distributions based on mean and variance, the more samples that are used, the more accurate the metric will be. This can be problematic with smaller datasets leading to an inaccurate representation of the GAN's performance.

Evaluating one GAN architecture with a metric such as FID is a valid method to compare different hyperparameters and small tweaks in architecture. This can be done by generating many synthetic images from each configuration and calculating the respective FID values. Based on these values a conclusion as to which model performs best can be reached. When comparing results between different GAN architectures trained on different data this is unfortunately not the case. Comparison of the minimum FID is a meaningless metric which does not accurately represent the difference in the performance of two GANs. One suggested method to correctly evaluate models is comparing the distribution of the FIDs over a fixed sample size with a fixed computational budget [30].

In many cases, a purely visual inspection is the best indicator of GAN performance. To perform a qualitative analysis a human actor is needed in order to assess the image quality. As done by [39] they asked a group of human annotators to distinguish between real and generated data. Unfortunately, they found that the motivation of the annotators varied and tainted the results from this type of evaluation. Another interesting finding is that by giving the annotators feedback about whether they correctly labeled an image improved their ability to spot the fakes as more images were shown. So as more images were evaluated, the quality of generation seemingly went down due to this fallacy.

2.3 Extensions of the GAN

Since the introduction of the GAN in 2014 [11] several new extensions and variations to the original architecture have been proposed. The original GAN is limited by only being able to generate a random sample from its training distribution. To change this, the authors of the Conditional GAN (CGAN) [32] present a method to include a class label into training. By conditioning each data sample on a class label it gives the ability to control which class will get generated during inference. The CGAN loss builds on the minimax loss as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_{data}(x)} \left[\log D(x|y)\right] + \mathbb{E}_{z \sim p_z(z)} \left[\log(1 - D(G(z|y)))\right]. \tag{2.4}$$

where the discriminator D and generator G model the conditional probability given a class label y. Aside from this, the loss is identical to the original minimax loss.

The downside from this method is the training data now requires a class label for each data input as this GAN. Even though this labeled data may be difficult and time-consuming to obtain, the advantages with CGAN is a powerful improvement because generating random samples as done by traditional GANs might not be useful in certain applications.

The convolutional neural network (CNN) has long been among the best performing models in image recognition tasks. Borrowing the operation of convolutions and incorporating them into generative adversarial networks shows a great increase to performance on image generation tasks. Introduced by Radford et al., [37] the deep convolutional GAN (DCGAN) is a new GAN architecture that is specialized to improve image generation by replacing fully connected layers with convolutions in both the generator and the discriminator networks. Guidelines that are proposed include: use of strided convolutions in the discriminator and fractional-strided convolutions in the generator, use of batch normalization in both networks, remove all fully connected layers, use ReLu activation in the generator except for output layer which uses Tanh, and use LeakyReLu in the discriminator. These design choices result in more stable training and the ability to support higher resolution generation than previous GAN networks. The success of the DCGAN proves the importance of using convolutions in GANs and has become standard across most novel GAN architectures used in computer vision tasks.

Progressive GAN (ProGAN) [19] introduces stability and decreased training times in generating higher resolution images. Karras et al. introduce a new method of progressively growing the size of the generator and discriminator during training. ProGAN begins with generating low-resolution images that are 4x4 resolution and throughout the training process additional layers are continuously added onto the generator and discriminator until 1024x1024 resolution images are being generated. Introducing these layers progressively was found to greatly increase stability in training. This is due to correctly generating a small image is easier than generating a large one. Instead of trying to generate higher resolution images from scratch, the ProGAN correctly learns the features of the image in a lower resolution first. Afterwards, slowly fading in the higher resolution layers in both the generator and the discriminator slowly builds the image at a higher resolution providing more accurate images and more reliable training patterns.

StyleGAN [21] is another variation of generative networks that expands on the ProGAN and gives users more control over the generation of images and being able to produce high-resolution images. In their paper, the authors introduce an alternative generator architecture while keeping the discriminator true to the original GAN architecture. Instead of inputting a random noise vector to the generator, the generator will begin training from a learned constant which gets adjusted with each convolution. This learned constant is provided by a mapping network that along with adaptive instance normalization (AdaIN) [13] provide a style to each convolutional layer. The mapping network is a eight-layer feed-forward neural network that transforms the noise vector into a 'style' vector which better represents the features the GAN is trying to learn. AdaIN is a normalization technique which includes a scale and translation factor which is what the authors declare as giving control over the style of the generation. AdaIN in expressed as:

$$AdaIN(x_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i}. \tag{2.5}$$

where each feature vector x_i in the convolutional layers gets normalized individually. As well as being scaled and translated by the style of y. The style is the latent vector containing characteristics learned by the mapping network.

Additionally, every convolutional layer gets some extra noise as input in order to introduce stochastic variation. The authors generate high-resolution human faces and explain how introducing this noise helps the network generate the stochastic aspects of human faces such as hair and freckles. Due to this noise being added to every layer, there is no carryover from previous layers which creates a localized application of this stochastic noise.

StyleGAN2 [22] is a direct extension of the first styleGAN model that fixes common issues that were discovered. First, it was found that the AdaIN operation caused water droplet characteristics during generation and it was fixed by redesigning the normalization. Weight demodulation is introduced which normalizes the weights in each convolutional layer directly and removes the need for AdaIN. Weight demodulated normalized weights w''_{ijk} are defined by:

$$w''_{ijk} = w'_{ijk} \Big/ \sqrt{\sum_{i,k} {w'_{ijk}}^2 + \epsilon}. \tag{2.6}$$

where w'_{ijk} represents the scaled convolution weight i with corresponding feature maps j and size of the convolution k and where ϵ represents a small constant.

Additionally, the method of progressive growing is re-visited and found that replacing it with skip connections between low and high-resolutions resulted in better generation. The skip connections allow the network to learn which resolutions are important for the final generated image and it was found that the network automatically applies a method similar to progressive growing without being told to. The final generation is a summation of all the lower resolution images with their respective contributions.

StyleGAN2-ADA [20] introduces data augmentation to help alleviate overfitting in the discriminator of the StyleGAN2 model. When small amounts of data are used to train GANs the discriminator can easily overfit due to easily being able to detect the small amount of real images. An important insight from the paper is the idea of leaking augmentations. Leaking augmentation can be experienced if the generator network 'sees' the augmented images and starts to learn to produce these altered images. In their paper they use a total of 18 image altering techniques in order to implement the discriminator augmentation. Early testing showed that in some cases higher levels of augmentation became harmful to the training. This means the amount of augmentation applied would need to be reasoned forward depending on the dataset. In order to get rid of this unnecessary hyper parameter adaptive discriminator augmentation was implemented. Adaptive discriminator augmentation (ADA) makes this tuning of augmentation dynamic so it automatically adjusts the probability of applying augmentation as the network trains and can increase or decrease based on if the network starts to overfit. The idea is that the amount of augmentation slowly increases as the GAN has seen the images multiple times over many epochs to help stop overfitting.

Another consequential improvement seen by the introduction of ADA is the possibility to train GANs and achieve good results with only a few thousand training images. With the previously introduced GAN architectures the amount of training images is orders of magnitude more than what is needed with styleGAN2-ADA.

Style Transfer

Style transfer is the process of altering some image A by applying the style from another image B onto the content of A. Using convolutional neural networks (CNNs) Gatys et al. [9] show how the recent improvements to CNNs can extract semantic information from images. Their method, A Neural Algorithm of Artistic Style, works by using feature representations from convolutional layers that capture textures and style from an image. This style can then be applied to the content in a new image. This method of style transfer is limited by being able to extract style from only one image. To get true style transfer from one whole domain to another, generative adversarial networks can be used.

Using the findings from conditional GANs, the creators of the pix2pix [15] network in their paper Image-to-Image Translation with Conditional Adversarial Networks introduce a GAN model to transfer styles from one domain to another. Pix2pix learns by conditioning the input image that the GAN is trying to generate with an additional image containing the style the synthetic image should have. The generator network is a convolutional U-net based architecture with skip connections which help the network correctly generate scenes in correct locations. The discriminator includes a convolutional patch-GAN classifier. This means that instead of evaluating a whole image at once in order to determine if it is real or fake, a patch-GAN classifier splits the image into smaller patches and decides whether each patch is real or fake. By averaging the patch results a final decision can be made for the whole image. This is advantageous as the Patch-GAN classifier is faster, yet still produces high quality results. While the results from pix2pix are impressive, the model requires paired input data which is still a difficult task in many domains. In most cases creating these training pairs between images can be expensive and time-consuming, and in some cases it might not even be possible as the expected output may be uncertain. This issue disappears with a model capable of learning the translation from one domain to the other completely by itself. This is exactly what is done by using two GANs combined into one large model introduced in 2017 called CycleGAN [47].

Cycle-consistent Adversarial Networks (cycleGAN) is a large model containing four networks. Introduced in their paper, Jun-Yan Zhu et al [47] describe a new GAN architecture that is capable of translating styles from one image onto another image without the need for paired training examples. The importance of not needing paired training examples is a huge breakthrough. The solution to this is what the authors of cycleGAN introduce as cycle consistent translation. This can be directly compared to a language translation task where a sentence is translated from Swedish to English, and then back to Swedish. Being cycle consistent would mean the exact same sentence should be returned back. The same idea is true for cycle-consistent GANs and they introduce a cycle consistency loss to be able to effectively learn unpaired image-to-image translation. This loss calculates the information lost when translating from domain A to B, and then back to A from B. Given two domains X, Y CycleGAN contains two mapping (generator) networks G and F that learn the mappings $G : X \rightarrow Y, F : Y \rightarrow X$. Each of these networks employ the original minimax loss expressed in equation 2.1 where the discriminator and generator learn as adversaries. The addition of cycle consistent loss is introduced as:

$$L_{cyc}(G,F) = \mathbb{E}_{x \sim p_{data}(x)} \left[||F(G(x)) - x|| \right] + \mathbb{E}_{y \sim p_{data}(y)} \left[||G(F(y)) - y|| \right]. \tag{2.7}$$

where the generator G first brings data x into domain Y, then generator network F translates $G(x))$ back into domain X. Then the loss is calculated as the difference between this reconstructed data vs real data x. The same procedure is then replicated in the opposite direction. The full objective for cycleGAN adds the adversarial loss and the cycle consistency loss.

CycleGAN can be difficult to train as it consists of four different networks, two generators and two discriminators. In their paper, the authors mention the difficulties of training the cycleGAN model and state the complexity from one domain to other must be similar in order to get meaningful results, ie. the model cannot learn to turn houses into dogs, whereas, dogs into wolves would be feasible.

Super Resolution

Super resolution is the process of estimating and interpolating a high-resolution image from a low-resolution image. Similar to many novel GAN techniques that aim to improve already existing solutions, super resolution first achieved state of the art results using deep learning with CNNs [6] [16].

Super resolution GAN (SRGAN) [28] builds on the previous works done with CNNs by including an adversarial learning process that helps guide the model into generating more realistic super resolution images. The generator in the SRGAN is a deep neural network containing residual blocks with skip connections and convolutions that up-sample the input. The input into a SRGAN is a low-resolution image as opposed to random noise as done by other GAN architectures. The discriminator classifies images by using a large CNN in a similar manner to how DCGAN does [37]. With eight strided convolutional layers and a final sigmoid layer the network aims to determine if a given sample is a super resolution version of the image or the original high-resolution image. The perceptual loss for super resolution (SR) is then calculated as a sum of the content loss X and the weighted adversarial loss GEN:

$$L^{SR} = L^{SR}_{X} + 10^{-3} L^{SR}_{GEN}. \tag{2.8}$$

where the adversarial loss is defined as:

$$L^{SR}_{GEN} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR})). \tag{2.9}$$

The probabilities over all the training samples are minimized where $G_{\theta_G}(I^{LR})$ is the synthetic super resolution image generated from a low-resolution (LR) image that D_{θ_D} classifies.

The content loss is the information lost when comparing the generated image to the original high-resolution image and is calculated as the euclidean distance between feature representations after activation layers in a pre-trained VGG19 [41] (a deep CNN) network expressed as:

$$L^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2. \tag{2.10}$$

In the context loss $\phi_(i,j)$ are the feature representations at the j-th convolution and i-th max-pooling layer and W/H are the respective dimensions. The euclidean distance between these two representations of high-resolution and low-resolution can then be calculated as the context loss.

Enhanced SRGAN (ESRGAN) [44] expands on SRGAN by building a deeper model using residual-in-residual dense blocks which contain multiple skip connections and removes batch normalization layers. It also employs the relativistic GAN's [17] loss function to improve the GAN's learning process instead of the adversarial loss introduced in SRGAN. The context loss is also improved by calculating the distance between features before the activation as it was found that before the activation layer many more neurons were activated leading to a better representation of the feature maps. ESRGAN is capable of reaching state of the art performance in super resolution for sharpness and detail.

Evaluating the results in a super resolution task is commonly done by using a peak signal to noise ratio (PSNR) calculation to calculate the reconstruction quality for images which in this case would be a super resolution image versus the original high-resolution image. PSNR is a ratio between a signals maximum strength and the power of the signal's noise and is calculated as:

$$PSNR = 20\log\left(\frac{255}{\sqrt{MSE}}\right). \tag{2.11}$$

where 255 is the maximum signal strength of a pixel in an image and MSE is the mean squared error between the pixels in the original image and the super resolution image. The PSNR is then reported as decibels where a higher value represents a higher quality image.

2.4 Applications of GANs and Related Work

There are many interesting and diverse applications of generative adversarial networks. The ability to generate new synthetic data can be applicable in all domains that rely on data to train machine learning models. The control of generation given by Conditional GANs help make GANs a useful tool in a machine learning pipeline. One such use case is when data augmentation techniques do not provide enough of an increase to the data amounts in order to get the desired results. Using a GAN to generate synthetic data is a technique that can be used to help increase the amount of data of the underrepresented class in an unbalanced data distribution. In remote sensing, this can be applied in any application where deep learning models rely on large amounts of data such as object detection, segmentation, or classification.

Style transfer between images also has many diverse applications. The pix2pix model has great capabilities in image-to-image translation including translating winter scenes to summer scenes, sketches to photographs, photos to cartoons, and plenty more domains shown in their paper [15]. The applications in remote sensing is similar to generating new data, but instead of just randomly generating synthetic data, style transfer allows for direct control of the result. A researcher might be trying to train a model to detect forest fire damage from satellite imagery but has few images containing damages. A GAN could potentially learn the mapping from healthy forest to burned forest and therefore be able to produce synthetic data to improve the detector.

Resolution is an important topic in remote sensing. In most applications, higher resolution imagery is more desirable than low-resolution imagery. The higher the resolution the more expensive the data due scarcity of satellites or UAVs that have larger sensors or the inability to capture imagery closer to the area of interest. One solution to this issue would be to capture low-resolution imagery and then use a GAN for super resolution to create synthetic imagery at high-resolution.

Data Augmentation with GANs

In a paper by Antoniou et al [1] they explore Data Augmentation using Generative Adversarial Networks. Their findings indicate that using a GAN to automatically augment data increased classifier performance. They introduce a new architecture they call Data Augmentation GAN (DAGAN) that is specifically used to simulate classical data augmentation. Their model takes data from one domain and with some data item, it can learn to generalize this data item to be able to generate similar data. This was proven in their results to be effective in many applications such as standard classifiers and matching networks.

Using generative adversarial networks [45] augmented a dataset in order to detect tomato disease in images of plants. Their initial problem is one similar to many machine learning projects: having an uneven distribution of data on top of not having enough data overall. Their classifier models were struggling to detect disease in tomato plants in China. Using a GAN they successfully trained a DCGAN in order to generative synthetic images of plants that contained the rare disease they were trying to detect. Re-training their original models with the GAN augmented data, they received significant improvements proving the usage of GAN generation in a machine learning pipeline.

Style Transfer with Satellite Imagery

In a 2018 research paper success in generating fake satellite imagery using generative adversarial networks was published in order to help show the potential of spoofing satellite data. Xu et al [46] using the CycleGAN [47] architecture successfully transfer the style of the cities of New York, Seattle, and Beijing onto basemaps. The style would be the architecture of a city from a satellite image perspective and a basemap is a reference map that shows topography of features like buildings, streets, and parks but omits realistic details. Their method involves training cycleGAN on unpaired images of basemap tiles and satellite images from the specified city each of which are 512x512 pixels in size. Their results show that the CycleGAN model can sufficiently apply styles of different cities onto basemaps.

Cloud Removal

Remotely sensed satellite imagery often contain clouds that obstruct the area of interest. There are techniques to remove clouds by simply retrieve multiple satellite images over the same area and replace the obstructed pixels from one image to the other. By implementing a cycle consistent [47], Cloud-GAN [42] removes thin clouds from satellite imagery by learning a mapping from cloudy to cloud-free imagery. Cloud obstruction can be categorized into thin cloud cover which partially obstructs information and thick cloud cover that completely obstructs objects. The limitations in their findings is their GAN can not successfully remove thick cloud cover as the loss of information is too large and they propose usage of higher wavelength data that can penetrate cloud cover such as synthetic-aperture radar (SAR).

By using SAR data Gao et al. [8] created a GAN model that successfully could remove thicker clouds from satellite imagery. Their proposed method uses a CNN to fuse together RGB and SAR images which are then used to learn a mapping from cloud covered imagery and cloud-free imagery with a GAN model based on the pix2pix [15] architecture. While their proposed method outperformed CNNs and GANs in cloud removal, there are clear limitations in obtaining SAR data. They propose that the solution to successfully remove thick cloud cover is temporal data in order to correctly reconstruct the information that is obstructed by the clouds.

Super Resolution

Using ERSGAN Pashei et al. [36] analyze the capabilities of super resolution in remote sensing. Their data consists of UAV captured imagery over a residential area in Texas affected by hurricane damage. By down-sampling their high-resolution imagery they created artificial low-resolution data in order to train an ERSGAN model for super resolution. They test their results in photogrammetry software that creates orthomosaics from drone imagery. Their results show no significant loss of information between the ground truth high-resolution and synthetic super resolution images reported by the software used. Furthermore, the synthetic super resolution imagery exhibit the same characteristics as the original high-resolution imagery in qualitative and quantitative assessments.

3 Method

This chapter presents the approach that is taken to answer the proposed research questions. With the intention of exploring a variety of different applications of GANs in remote sensing, a series of four distinct experiments are implemented. Each experiment number is connected to the corresponding research question that it tries to answer.

- Experiment 1: an evaluation of the generation capabilities of GANs using DCGAN and StyleGAN2-ADA trained on UAV imagery.
- Experiment 2: testing the capability of paired image-to-image translation from segmentation masks to satellite imagery using the pix2pix architecture.
- Experiment 3: an unpaired image approach to remove thin cloud cover and haze from satellite imagery using CycleGAN.
- Experiment 4: evaluating ESRGAN for the task of super resolution using satellite imagery.

The experiments performed are split into sections that each contain relevant information about the data, pre-processing steps, implementation, and training details.

These four evaluations are meant to cover a variety of tasks that are applicable in the field of remote sensing where GANs can be employed. The evaluation of each experiment will be done primarily with a qualitative assessment of image quality and usability. The details of the evaluation procedures are outlined in the final section of this chapter.

3.1 Experiment 1: Generation

To sufficiently answer the first research question regarding what resolution and quality of aerial imagery of forests GANs can generate, two approaches are implemented. First, the DCGAN architecture is implemented for the generation of low-resolution images, then the StyleGAN2-ADA architecture is implemented to generate higher-resolution images. The DCGAN model is useful as a comparison metric showing the progression in resolution and quality of GAN generation. This experiment aims to train a model capable of generating synthetic data that resembles the characteristics of the training data. In remote sensing, the lack of data can be a hindering issue for any machine learning solution. Successfully training a GAN to produce synthetic remotely sensed data can create new data augmentation techniques through generation aiding in many remote sensing tasks.

Data and Pre-processing

The images used throughout experiment 1 are UAV captured aerial images of Swedish forests. Two drones are used in the collection of this data: DJI Phantom P4 and DJI Mavic Pro 2 with all imagery being captured in the visible light spectrum (RGB). The images are captured within summer months (May - September) and are all consistent with the same flight parameters. The images are captured with a vertical and horizontal overlap of 30% and the flight altitude is consistent at 110 meters in all images. UAV images are captured with an overlap between successive images meaning that the same area is present within more than one image. This is useful to create orthomosaics using photogrammetric software, but in this experiment, the 30% overlap leaves room for cropping the images without losing information.

Aerial images taken by UAVs suffer from distortion causing leaning entities around the edges of the frame due to the wide-angle of the lens. To help remove some of this distortion the center areas of images are cut out removing much of the tilt distortion that occurs along the edges of the image. This process helps keep data more uniform by having images containing trees that are generally more orthogonal. Afterward, patches of size 1024x1024 pixels are created from the center cutouts. Figure 3.1 demonstrates this process. This results in a total image count of around 1400 patches. The images are scaled appropriately to the desired output size before the training of each model. In this task of generation a test and train split is not used because the models are learning completely unsupervised to learn the underlying distribution of the training data and the metrics compare newly generated synthetic data to the original data.

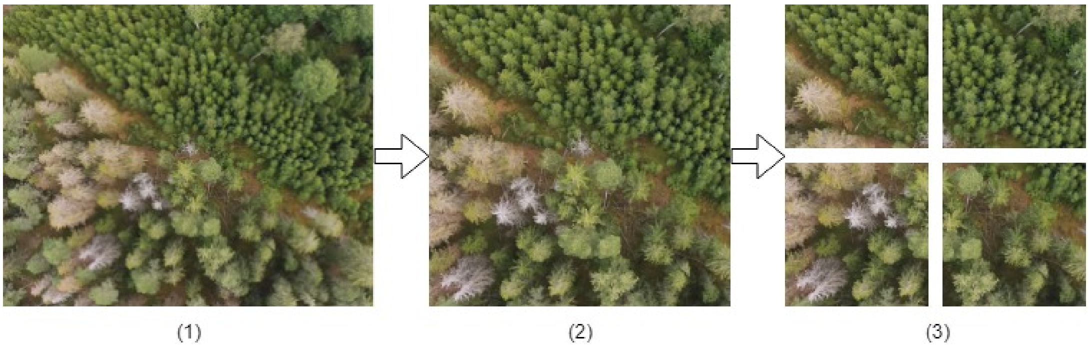

Figure 3.1: The original image (1) has its the edges evenly cropped resulting in the square center crop (2) which is then split into even patches (3).

Implementation DCGAN

The deep convolution GAN is implemented using Pytorch [1] to test the generation capability of a simple network that is limited to low resolutions. All the input data is scaled to be of size 64x64 pixels before training the model. The generator network receives as input a latent vector of length 100 which is fed into five convolutional layers which convert the vector into a three-channeled RGB image of size 64x64 pixels. The discriminator is a mirrored implementation of the generator with an additional final sigmoid layer that can classify if images are

[1] https://pytorch.org/

real or fake. The DCGAN implementation uses an Adam optimizer with beta coefficients of $\beta_1 = 0.5$ $\beta_2 = 0.999$, a learning rate of 0.0002, and is trained for 100 epochs on an Nvidia GTX 1660 GPU.

Implementation StyleGAN2-ADA

An official Pytorch implementation of StyleGAN2-ADA supplied by the Nvidia research team is used for generating high-resolution imagery. The model is available as open-source on Github [2].

StyleGAN2-ADA is trained on the same dataset of UAV images used to train DCGAN except kept at their original size of 1024x1024 pixels. Instead of training a randomly initialized network from scratch, transfer learning is used to reduce training times and improve results. The starting weights used are from a pre-trained model that has been trained to generate full HD human faces at 1024x1024 pixel resolution.

The training configuration uses random flip augmentation for every image in the training set doubling the size to contain 2800 images. The resolution of generation is increased from the default setting of 512x512 pixels up to 1024x1024 pixels. The Adam optimizer is used in both the generator and discriminator networks with a learning rate of 0.002, beta coefficients of $\beta_1 = 0.9$ $\beta_2 = 0.99$, and a batch size of 4. All other training parameters are consistent with the original styleGAN2-ADA implementation. The model is trained for 320 epochs on an Nvidia RTX 3090 GPU.

3.2 Experiment 2: Image-to-Image Translation

Experiment 2 tries to answer the second research question by implementing a GAN for paired image-to-image style transfer. The pix2pix model is used to learn the mapping from a segmentation mask to a real image using satellite imagery. By training a GAN to recreate a realistic image from a segmentation mask the result can be used as a controllable data generation technique. Similar to experiment 1 the synthetic data can be used in many remote sensing tasks where a large amount of data is needed. In many cases, the ability to control generation is a benefit over the random generation explored in experiment 1.

Data and Pre-processing

To be able to learn a mapping from segmented imagery to real imagery it is necessary to have paired data because pix2pix is a conditional GAN that learns through labeled image pairs. The data used contains segmentation masks of satellite imagery over rural landscapes in Poland. The segmentation masks are one-channel images where each pixel is given a value to classify the pixel in the corresponding RGB image. The classes in the segmentation masks are buildings, forest, water, and ground. The ground class also includes anything not captured by the other classes such as roads. The data is publicly available at LandCover.ai[3]. The original one-channel segmentation masks are converted into a three-channel RGB representation using QGIS[4] to allow for easier visualization during training of the pix2pix model. Figure 3.2 shows an example of a segmentation mask after the color has been converted and the corresponding satellite image. The full-size satellite images and masks are then split into patches of size 256x256 pixels using Python for a total of 1,024 patches. A split of 10% of the data is left as test data and 90% is used for training.

[2] https://github.com/NVlabs/stylegan2-ada-pytorch

[3] https://landcover.ai/

[4] https://qgis.org/en/site/

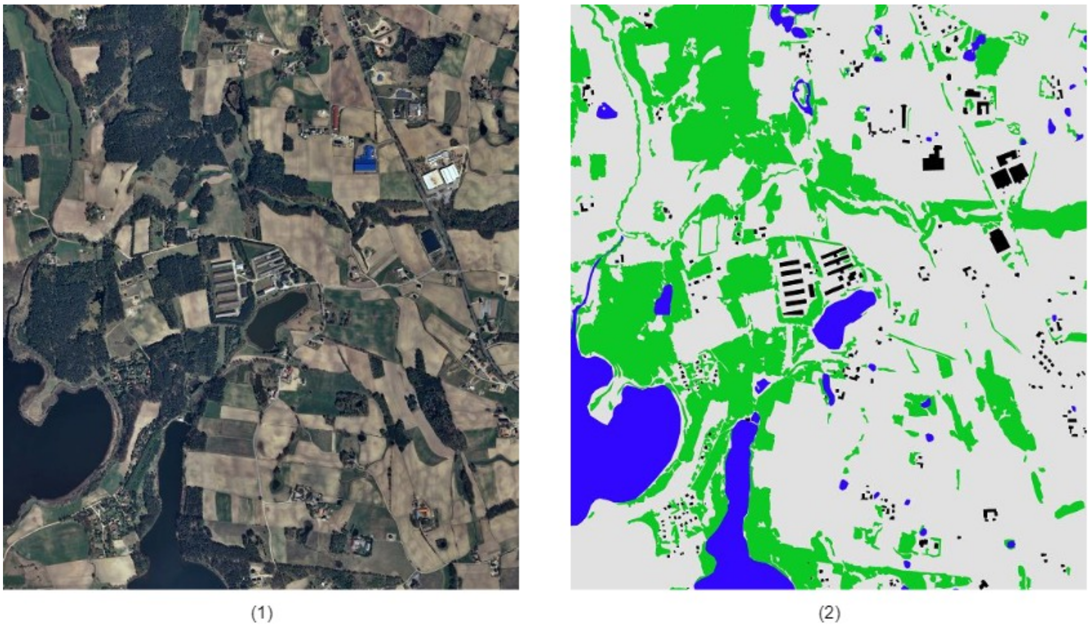

Figure 3.2: Satellite image (1) and the corresponding segmentation mask (2)

Implementation

The official Pytorch implementation of pix2pix was used for evaluating image-to-image translation. The model is available as open-source on GitHub[5].

Pix2pix was trained on paired images and the dataset structure is created with a script provided by the authors to concatenate each pair into one file containing the mask and the satellite image side by side. Using the default parameters defined by the original pix2pix model configuration of training image size of 256x256 pixels, the Adam optimizer with β_1 = 0.5 and β_2 = 0.999, a learning rate of 0.0002, using a batch size of 1, and training for a total of 200 epochs on an Nvidia RTX 2080 Ti.

3.3 Experiment 3: Cloud Removal

Experiment 3 attempts to answer the third research question regarding a GANs ability to remove clouds using unpaired image-to-image translation. In this experiment, CycleGAN is trained to remove thin cloud cover from satellite imagery. Cloud cover is a common issue when working with satellite imagery. Many times the area of interest in a remote sensing task is obstructed by clouds depending on location and time of year. Clouds in satellite imagery can be categorized into two types: thin and thick clouds. Thick clouds fully obstruct the area underneath them and leave a near-impossible task to remove them without the use of temporal or multi-spectral data. In this experiment, the thin clouds and haze-like obstruction are the focus as they only partially obstruct the area underneath them. The goal is to train a GAN to remove thin clouds from satellite imagery by generating synthetic cloud-free images.

Data and Pre-processing

For the cloud removal task, open satellite imagery from Sentinel-2 was used. Sentinel-2 is a satellite that was deployed by the European Space Agency in 2015. This satellite has global

[5] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

coverage and will re-capture imagery over every location about every ten days. The resolution of the satellite data is 10 meters meaning that every pixel in the image represents a real-world measurement of 10x10 meters. Sentinel-2 satellite data is available for download from the Copernicus API. [6]

Manually finding enough training data where satellite imagery contains thin clouds or haze is difficult. Therefore, a method using Perlin noise [7] is used to simulate thin cloud cover that can then be applied to satellite images that do not contain cloud cover. Perlin noise is an algorithm that has the ability to generate randomized textures and patterns which in this case will be configured to be visually similar to clouds. First, 1600 patches of size 512x512 pixels are created from a large Senintel-2 satellite image. Afterward, the simulated randomized Perlin noise pattern is applied to each patch using alpha blending to produce a transparent cloud-like coverage as shown in figure 3.3. Alpha blending is the process of adding an extra alpha channel to an image which allows for adjustment of the transparency. The transparency of each cloud-covered image also varies to introduce more diversity into the data.

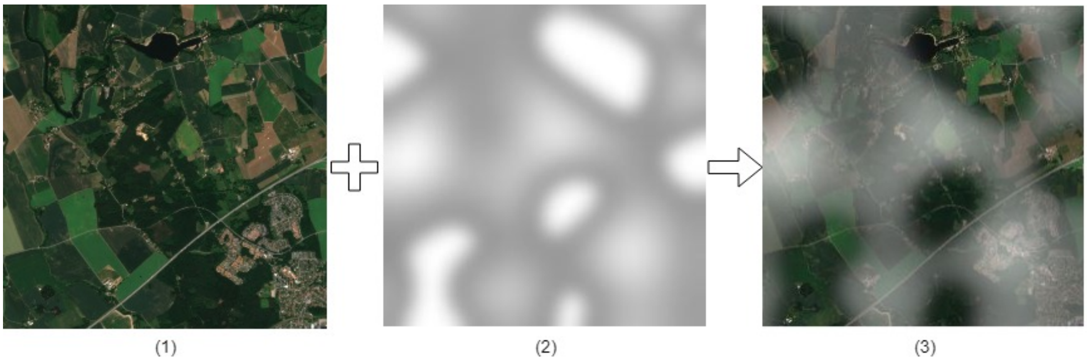

Figure 3.3: Using alpha blending the original satellite patch (1) plus the simulated perlin noise pattern (2) results in the final cloud simulated image (3).

Implementation

The official CycleGAN implementation in Pytorch was used for this experiment. The model is available as open-source on GitHub[7].

The training parameters for CycleGAN are left as the default configurations done by the original implementation except for increasing the default image size to 512x512 pixels. The training uses the Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$, a learning rate of 0.0002, a batch size of 1, and training for a total of 120 epochs on an Nvidia RTX 2080 Ti.

3.4 Experiment 4: Super Resolution

Experiment 4 answers the final research question regarding the ability to use a GAN to perform a super resolution task using satellite imagery. An implementation of ESRGAN is used to enhance the resolution of low-resolution images. The task to enhance the resolution of a remotely-sensed image is relevant in almost all applications of remote sensing. Having higher resolution imagery means smaller objects can be detectable and distinguishable from other objects. Even in tasks dealing with larger features higher resolutions lead to more accurate representations of shapes and areas in remotely sensed imagery.

[6] https://scihub.copernicus.eu/

[7] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

Data and Pre-processing

Satellite imagery from DOTA[8] is used for the process of evaluating the super resolution task. DOTA is a collection of remotely sensed imagery collected from Google Earth, and the GF-2 and JL-1 satellites. It is a popular open-source dataset used for segmentation and object detection tasks in satellite imagery. To train the ESRGAN model a low-resolution and high-resolution pair of images are required. In this case, the raw satellite images are used as the high-resolution images. In order to create the low-resolution pair a down sampling method using a bi-cubic kernel was implemented using OpenCV[9]. The down-sampling reduces the resolution by a factor of 4x. The original satellite images from DOTA are all very large non-uniform shaped satellite images that are not compatible with ESRGAN. To reduce the image size and make consistent data a pre-process patching is applied in order to make patches of 480x480 pixels for high-resolution images, and 120x120 pixels for the low-resolution images. This results in the pair of patches covering the same area but at different resolutions, as shown in figure 3.4. The data is further split into a training set containing 18,000 patches or about 90% of the data and 2,000 patches or 10% being used as a test set.

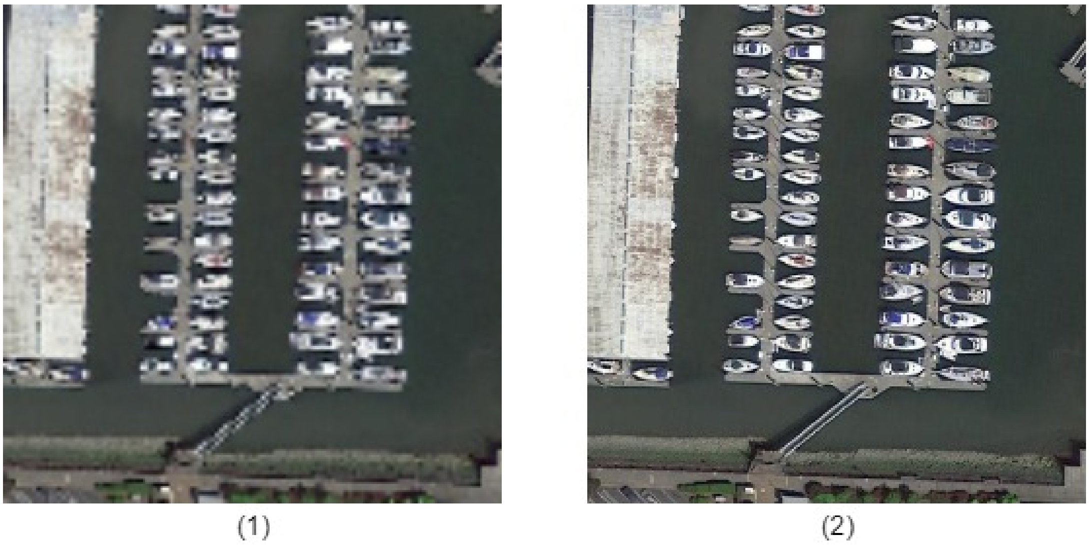

Figure 3.4: Example of a low resolution (1) and high resolution (2) image pair.

Implementation

The official ESRGAN Pytorch implementation was used in this experiment. The model is available as open-source on GitHub[10].

The configuration used is trained to increase the resolution of the input image by 4x. The training process of ESRGAN further splits the input data into smaller patches of 128x128 pixels as it was found by the original authors that it provides better performance and lowers training time. The generator and discriminator networks both use the Adam optimizer with β_1 = 0.9 and β_2 = 0.99, a learning rate of 0.0001, using a batch size of 16, and train for a total of 11 epochs training on an Nvidia RTX 2080 Ti.

[8] https://captain-whu.github.io/DOTA/dataset.html

[9] https://opencv.org/

[10] https://github.com/xinntao/BasicSR

3.5 Evaluation Metrics

There are two main methods that are used in the evaluation of generated imagery. A quantitative analysis using a mathematically calculated metric or a qualitative metric from a subjective human perspective using visual inspection. There are no metrics used to compare the different GAN models to one another because all four experiments are independent and they are trained to accomplish different tasks on different data sets.

In all experiments, a qualitative analysis of the generation is conducted by visually inspecting and evaluating the synthetic data. Results displaying good and poor quality of generation are reported and discussed. The visual inspection includes closely evaluating synthetic images and reporting any artifacts or flaws in generation, overall image quality, resolution, and the potential usability of synthetic data in the remote sensing task that is being evaluated based on all previously mentioned information. This is meant to be a thorough and unbiased analysis of both good and bad samples.

In addition to the qualitative evaluation, various quantitative metrics are reported to help convey the performance of the models and are used to help choose the best performing models:

- In experiment 1 the best DCGAN model is selected by monitoring loss values from the discriminator and generator. For the StyleGAN2-ADA model Fréchet inception distance (FID) is calculated from samples of generated images. To select the best performing model the FID is calculated at the end of each epoch using 50,000 images. After training, the model with the lowest FID is selected. After every epoch, a visual inspection of a batch of synthetic imagery is conducted to verify the visual improvement shown by the FID.

- In experiments 2 and 3 the best performing models are selected through visual inspection during training. After each epoch, a varying amount of images are generated and inspected visually to ensure improvement in the quality. The results are evaluated only with a qualitative measure comparing the synthetic images to the real training images. The loss values are not informative for selecting the best models as they oscillate heavily and do not provide enough insight into performance.

- In experiment 4 regarding evaluating super resolution, a peak signal to noise ratio (PSNR) is used as a metric to determine the best performing model. In addition to PSNR, a visual inspection of a batch of images after each epoch is done to verify quality and improvement and to help select the best model.

4 Results

This chapter presents the results following the experiments introduced in the method chapter. The results shown are picked to highlight a variety of good and poor quality generations and to display any important characteristics found in the results.

4.1 Experiment 1: Generation

The DCGAN model trained to 100 epochs taking around 15 minutes and the best performing model is chosen based on the loss values. The discriminator and generator loss begin to diverge at around 3000 iterations or 50 epochs which is a sign of overfitting therefore the weights at epoch 50 are used. The complete loss plot is available in the appendix in figure A.1. The model successfully generates aerial forest imagery at a 64x64 pixel resolution. The generated images are diverse in the way that mode collapse is not present but exhibit slightly worse quality when compared to the training samples. Further inspecting the results shown in figure 4.1 the real samples are much sharper and contain finer details at the individual tree level. Compared to the synthetic images which appear blurry and in some cases make it hard to distinguish individual trees. While generation is possible with DCGAN, the synthetic images are not particularly useful as a data augmentation technique in a remote sensing application due to their low-resolution.

The StyleGAN2-ADA model trained for 320 epochs taking around 6 hours and the FID metric for the best performing model is 13.925 which occurred at epoch 280. A full plot of the FID values throughout training can be seen in figure A.2. StyleGAN2-ADA has great capability of generating synthetic images at a much higher resolution than DCGAN at 1024x1024 pixels. In figure 4.2 a sample of synthetic images of good quality is presented. These images look realistic and have little to no artifacts or visible distortion upon inspection. The diversity is apparent and displays how the model did not simply learn to memorize a set of training images but rather learn how trees are represented and then generate new images based on the characteristics of the training data. This is further proven by the FID which would be closer to 0.0 if the GAN generated images were identical to the training images.

Three examples of poor quality results are shown in figure 4.3 where the generator produces images that contain distortion and unrealistic appearances. These types of images are

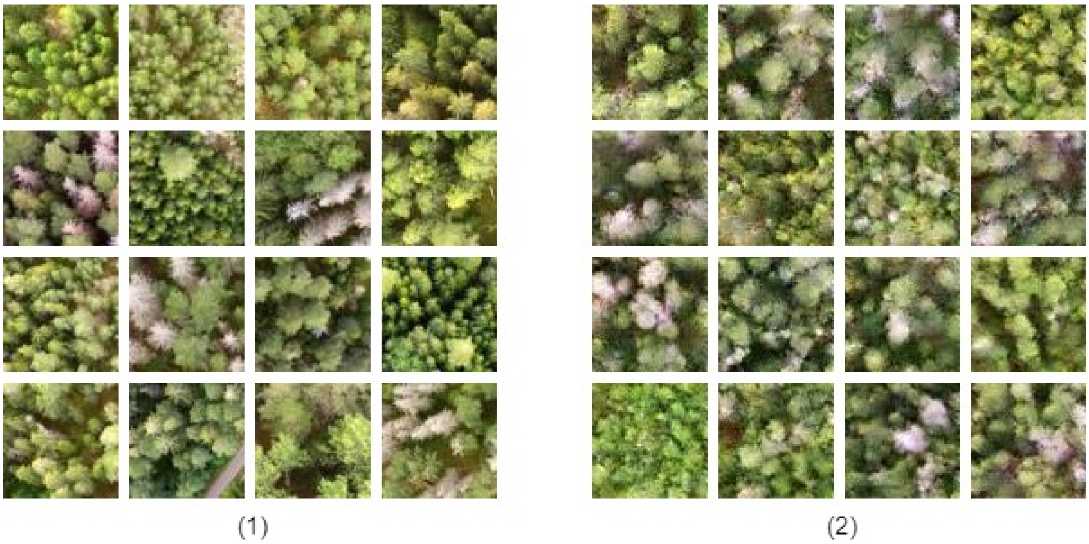

Figure 4.1: Random samples of real images (1) and DCGAN generated images (2).

uncommon in a batch of generated images but do occur. The main artifact seen are ripple effects where the ground and trees begin to look like water. Even with the distortion present the trees still look realistic with slightly worse quality due to some warping from the ripples.

The model has also begun to exhibit small amounts of mode collapse in the majority of synthetic images. As highlighted in figure 4.4 the areas shown in red have become consistently the same shape and color in some generations. This is an artifact of mode collapse as the training data does not contain this property.

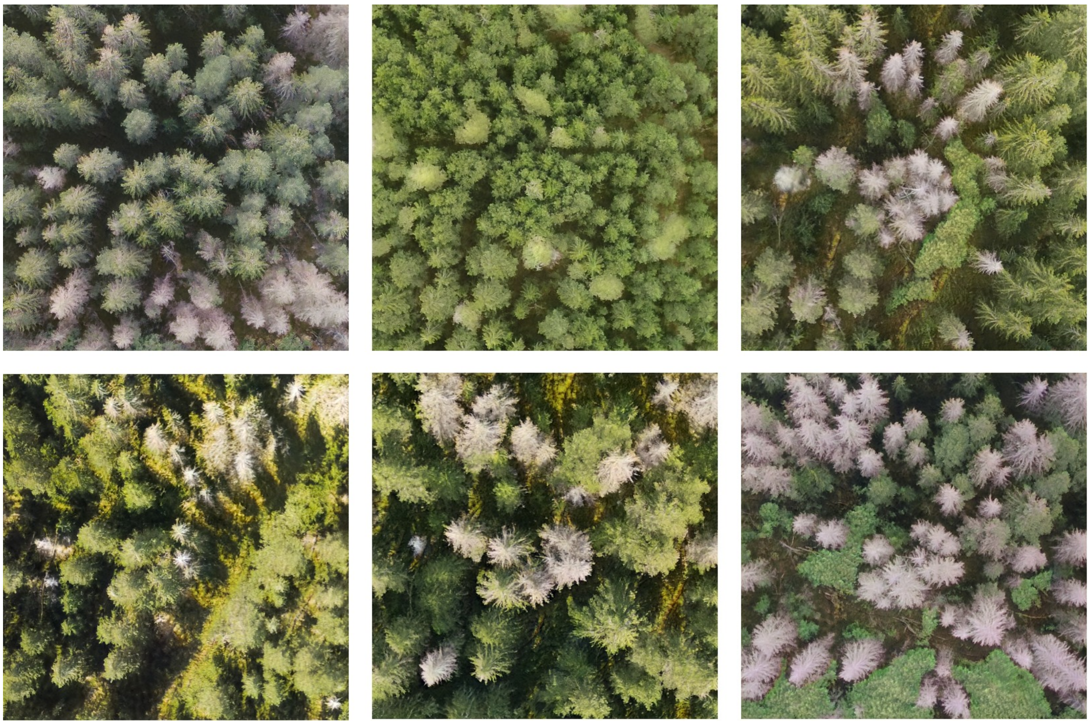

Figure 4.2: Samples of good quality StyleGAN2-ADA generated imagery

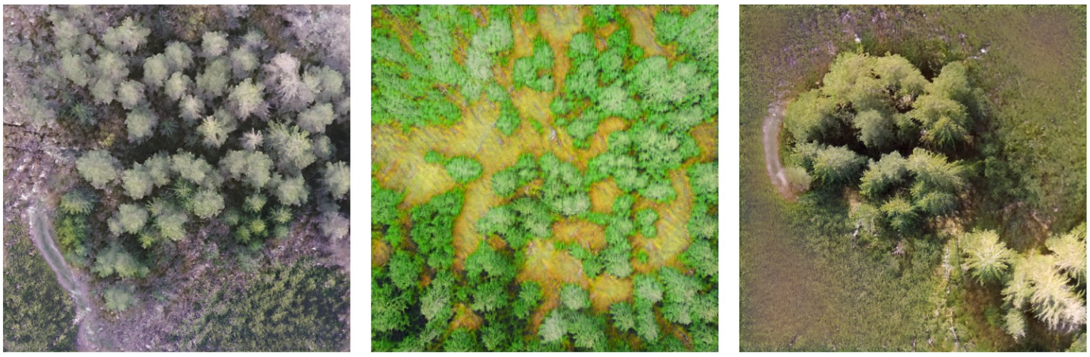

Figure 4.3: Samples of low quality StyleGAN2-ADA generated imagery

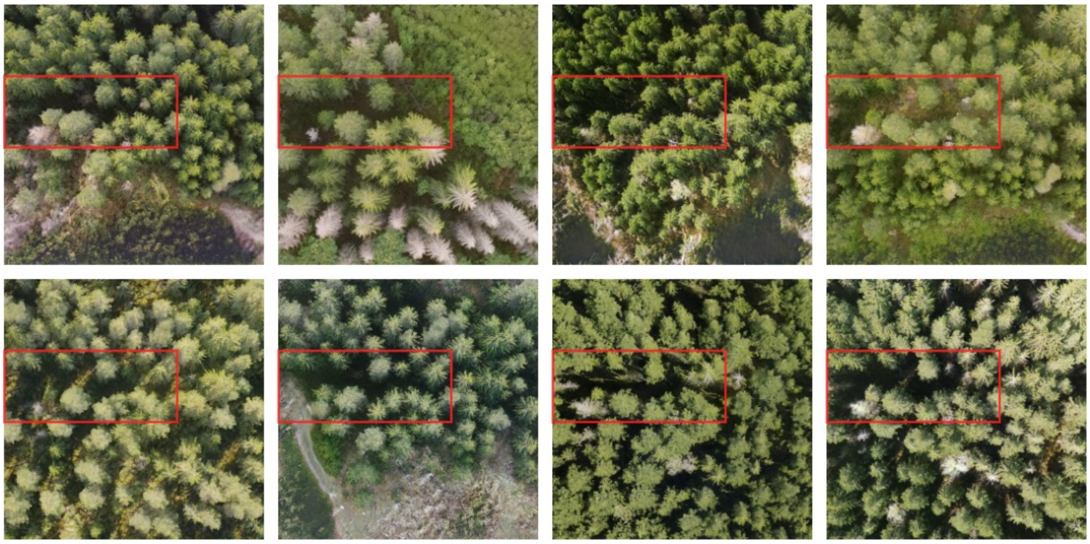

Figure 4.4: Highlight of mode collapse in StyleGAN2-ADA generated imagery

4.2 Experiment 2: Image-to-Image Translation

Pix2pix trained for 200 epochs taking about 2 hours. The best model is at 200 epochs and is selected by visually inspecting several of the generations from the final epochs and choosing the model that produces the most visually accurate images with respect to the training images. When checking the loss values not much information can be gained as they do not converge. The discriminator loss does begin to stabilize which shows the model was still learning up until the final epoch. The generator loss in figure A.3, discriminator loss for fake samples in figure A.5, and the discriminator loss for real samples in figure A.4 can be viewed in the appendix.

The samples shown in figure 4.5 display how the model failed to produce a synthetic image that looks realistic. These images show how the distortion interferes to a degree where large amounts of information is lost. It is difficult to decipher the green forest class from the ground in these images as the ground is quite randomly generated. On the other hand, the same model performed well on predicting some forests from the masks and some buildings as shown in figure 4.6. It can be seen that the model struggled most to learn that the grey color represents a general class of ground (anything that is not of class building, forest, or water). Yet, it performed well on the other classes where green is the forest, blue for water, and black represents buildings. In some cases even in these examples the synthetic data is not perfect. Distortion in the form of mild to severe blurriness is also present in all the images. The building class is also difficult to generate as the training data contains buildings with many colors of roofs. This information is not accurate in the synthetic images as seen by looking at the bottom-most example in 4.6.

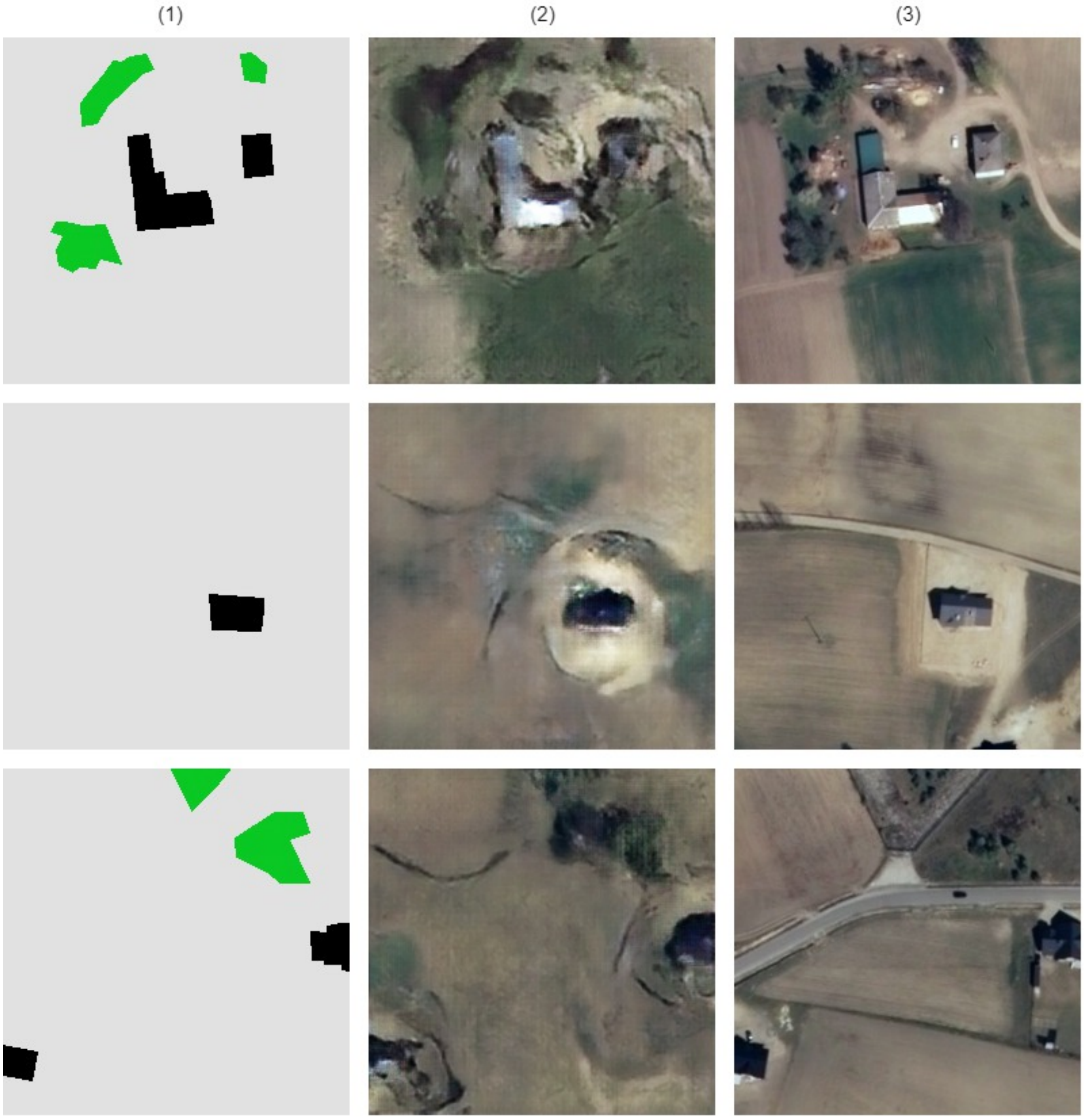

Figure 4.5: Samples of low quality pix2pix style transfers where the segmentation mask (1) is used as input to generate the synthetic image (2) and the ground truth image (3).

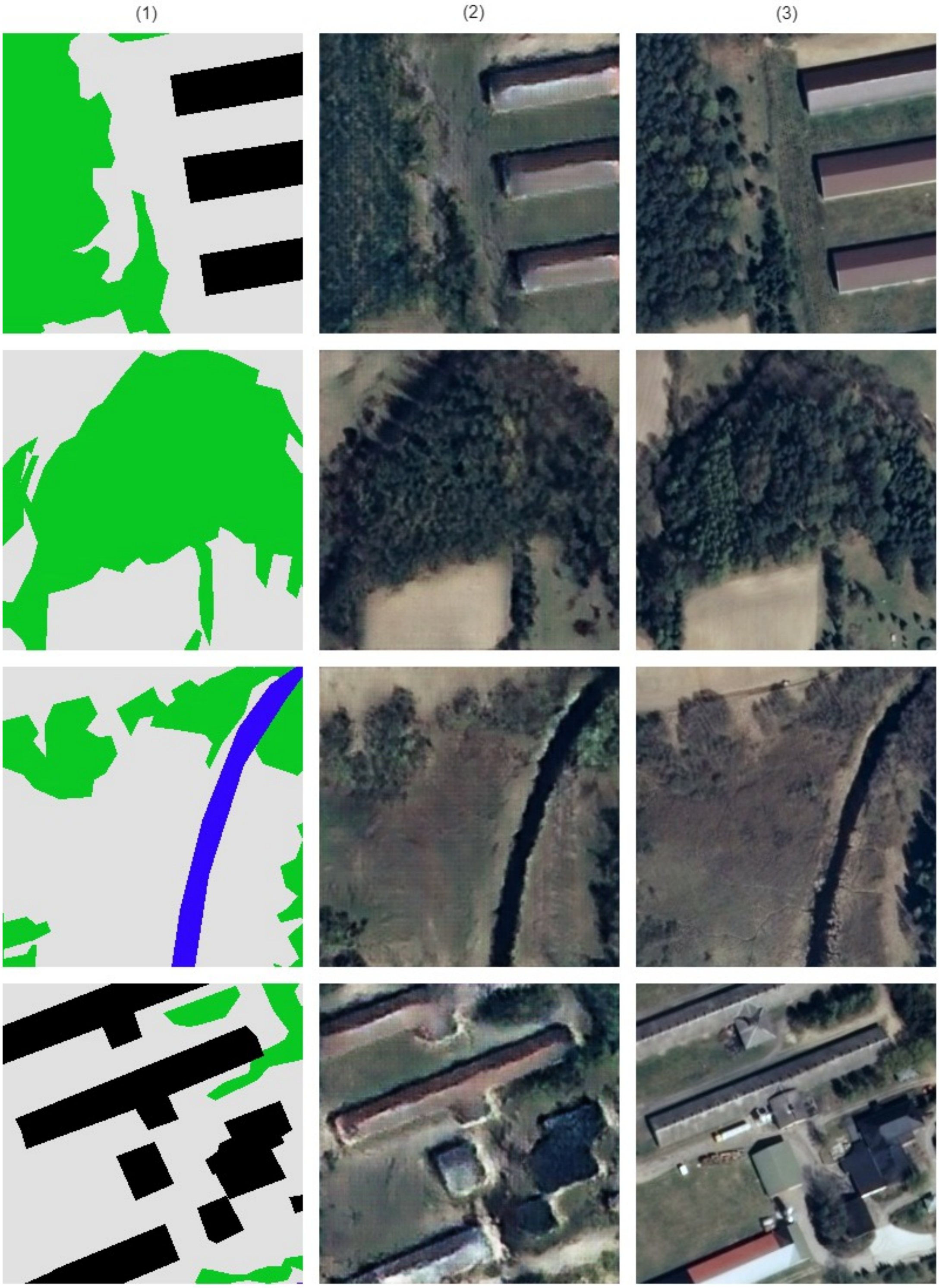

Figure 4.6: Samples of good quality pix2pix style transfers where the segmentation mask (1) is used as input to generate the synthetic image (2) and the ground truth image (3)

4.3 Experiment 3: Cloud Removal

After training for 120 epochs taking roughly 65 hours, CycleGAN successfully removes thin cloud coverage and haze like interference from satellite imagery. The final epoch was chosen as the best performing model through visual inspection at each of the last 60 epochs and based off the loss values which show that the model was not getting worse but convergence was not necessarily reached. The cycle consistent loss for the domain A (cloud) in figure A.6 and domain B (cloud free) in figure A.7 are available in the appendix.

Figure 4.7 shows four examples where the reconstructed image has little to no artifacts or noise introduced by the generation. The quality of the synthetic image is essentially indistinguishable from a real satellite image in the sense of information loss, sharpness, and overall color profile. When inspecting the regions located at the edges of clouds a small color change may be apparent, but in these samples the artifacts are barely visible.

On the other hand, figure 4.8 shows some examples of degradation during generation. In these cases, the model has over or underexposed areas that are not covered by clouds shifting the colors drastically enough to leave large artifacts on the image. Also, images containing water seem to fail more often than land covered areas in removing cloud coverage.

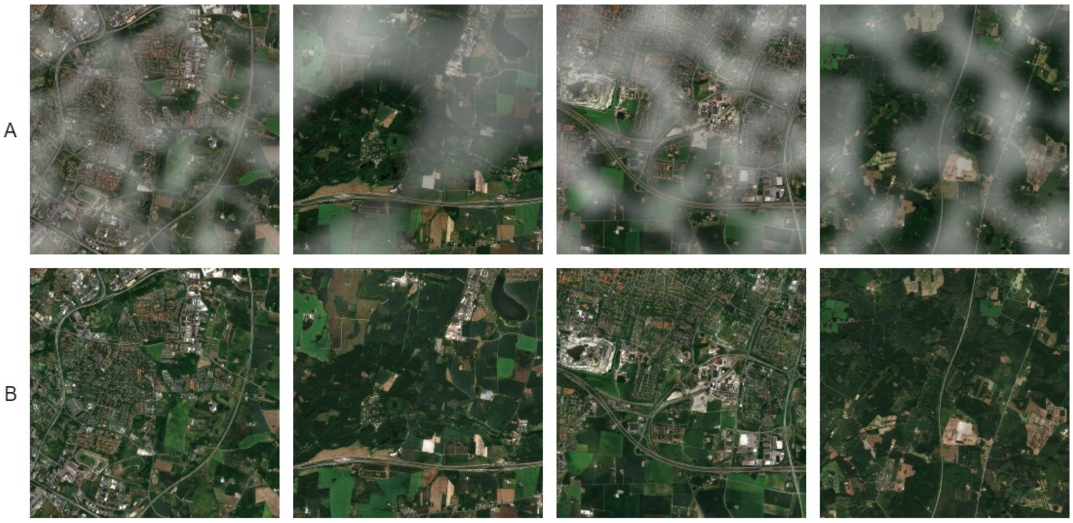

Figure 4.7: Samples of good mappings from domain A to B.

Despite the task being to remove cloud cover, CycleGAN contains two generators for learning the mapping of A to B and B to A. By using the generator that learned the opposite mapping this model is also capable of acting as a cloud simulator that can add clouds to remotely sensed imagery. While the cloud simulated images from CycleGAN are not especially useful in this study, they do exhibit some interesting traits such as generating clouds that follow water boundaries or clouds that follow paths of similar colors and brightness. A few examples of these traits are shown in figure 4.9.

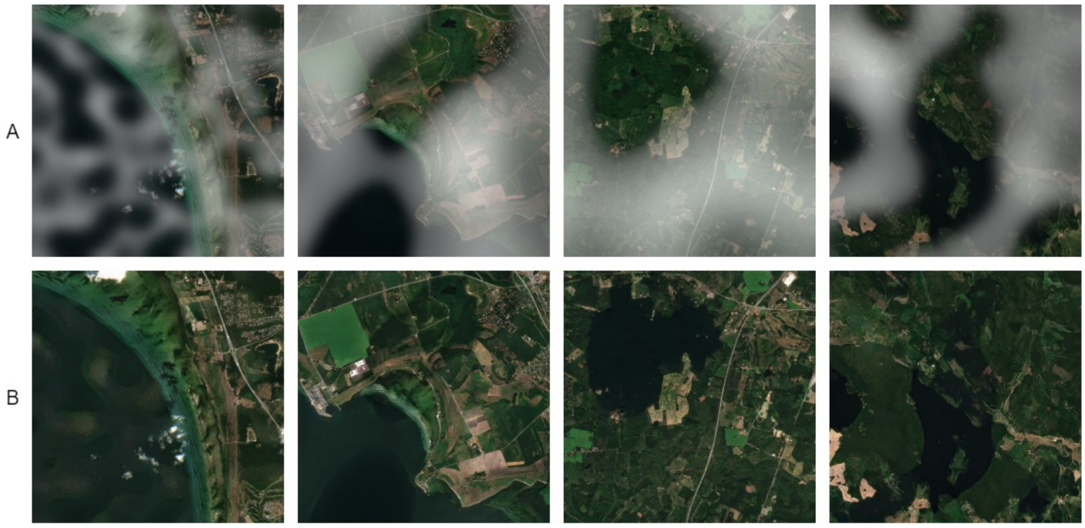

Figure 4.8: Samples of poor mappings from domain A to B.

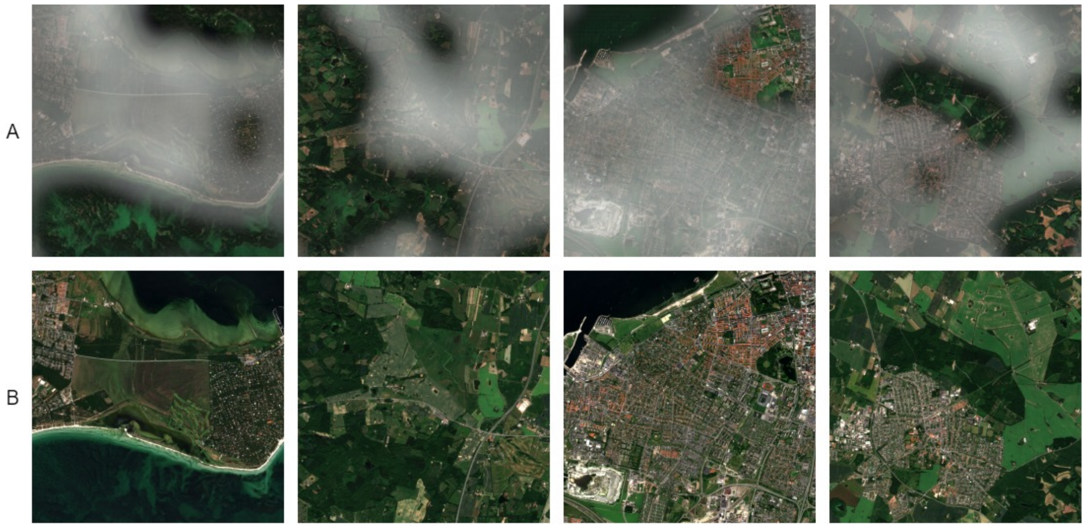

Figure 4.9: Samples of mapping from domain B to A.

4.4 Experiment 4: Super Resolution

ESRGAN was trained to approximate a high-resolution image from a 4x down-sampled image. Training 11 epochs took about 45 hours resulting in a PSNR of 19.4887 for the best performing model. The full PSNR plot is shown in figure A.8. The results shown in figure 4.10 present four samples of results each showing the low-resolution image, synthetic high-resolution result, and the ground truth image. The super resolution quality is very good with only minor artifacts. Figure 4.11 show two examples of such artifacts where the information lost in the low-resolution image made it impossible to up-sample into a correct representation. This leads to small features sometimes blending into each other, or becoming a blob-like shape instead of taking the shape of the actual feature. These distortions occur with small objects in the image that are close to each other because when converted into a low-resolution representation the features that were defined by multiple pixels become represented by just one pixel. From one pixel interpolating back into multiple pixels is what can cause these features to blend into each other.

Overall, the ESRGAN model for super resolution in satellite imagery tasks can be used to increase the spatial resolution for remotely sensed imagery granted accessibility to the necessary data.

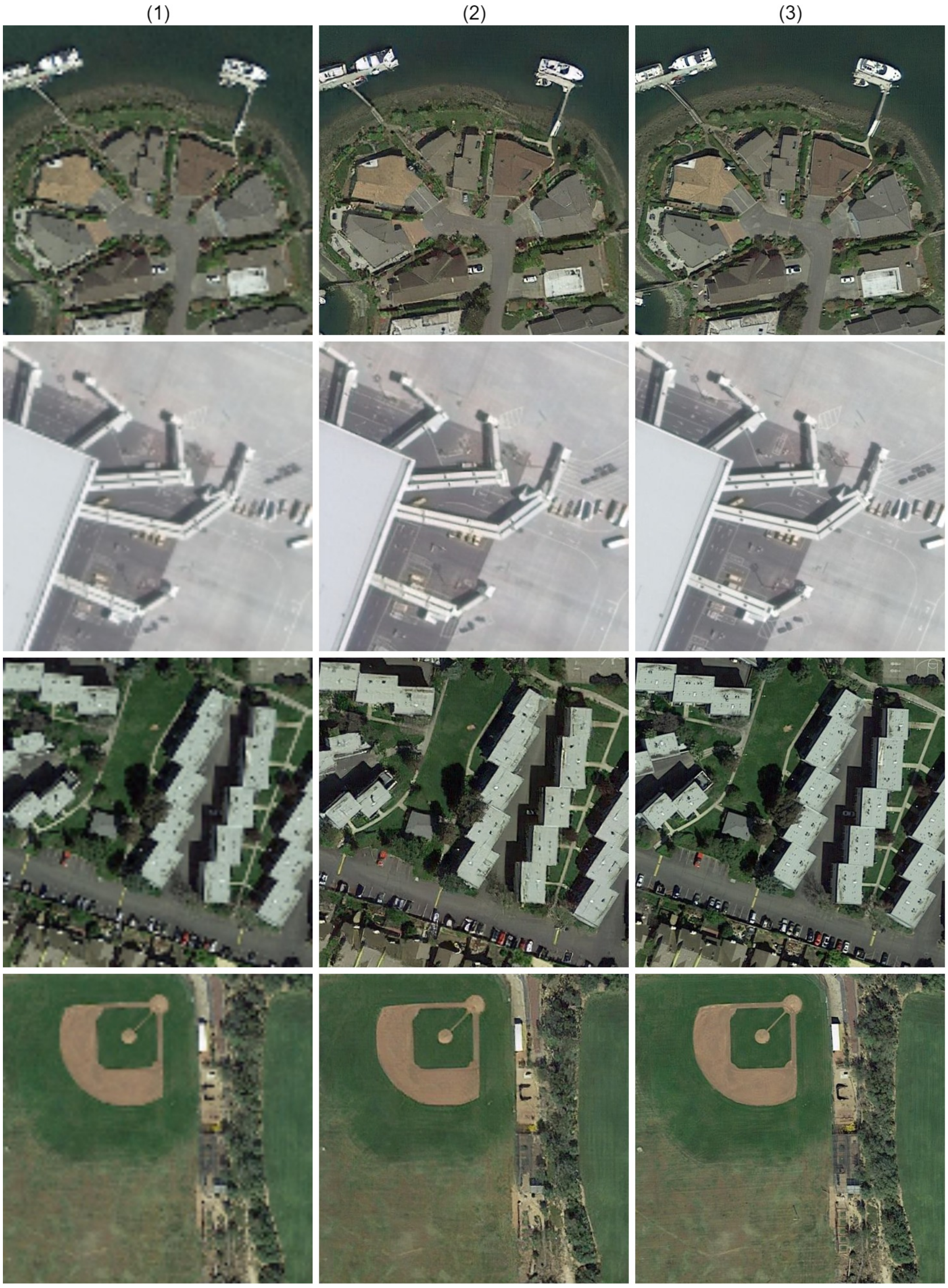

Figure 4.10: Samples of super resolution showing the low-resolution image (1), the synthetic high-resolution image (2), and the high-resolution ground truth image (3).

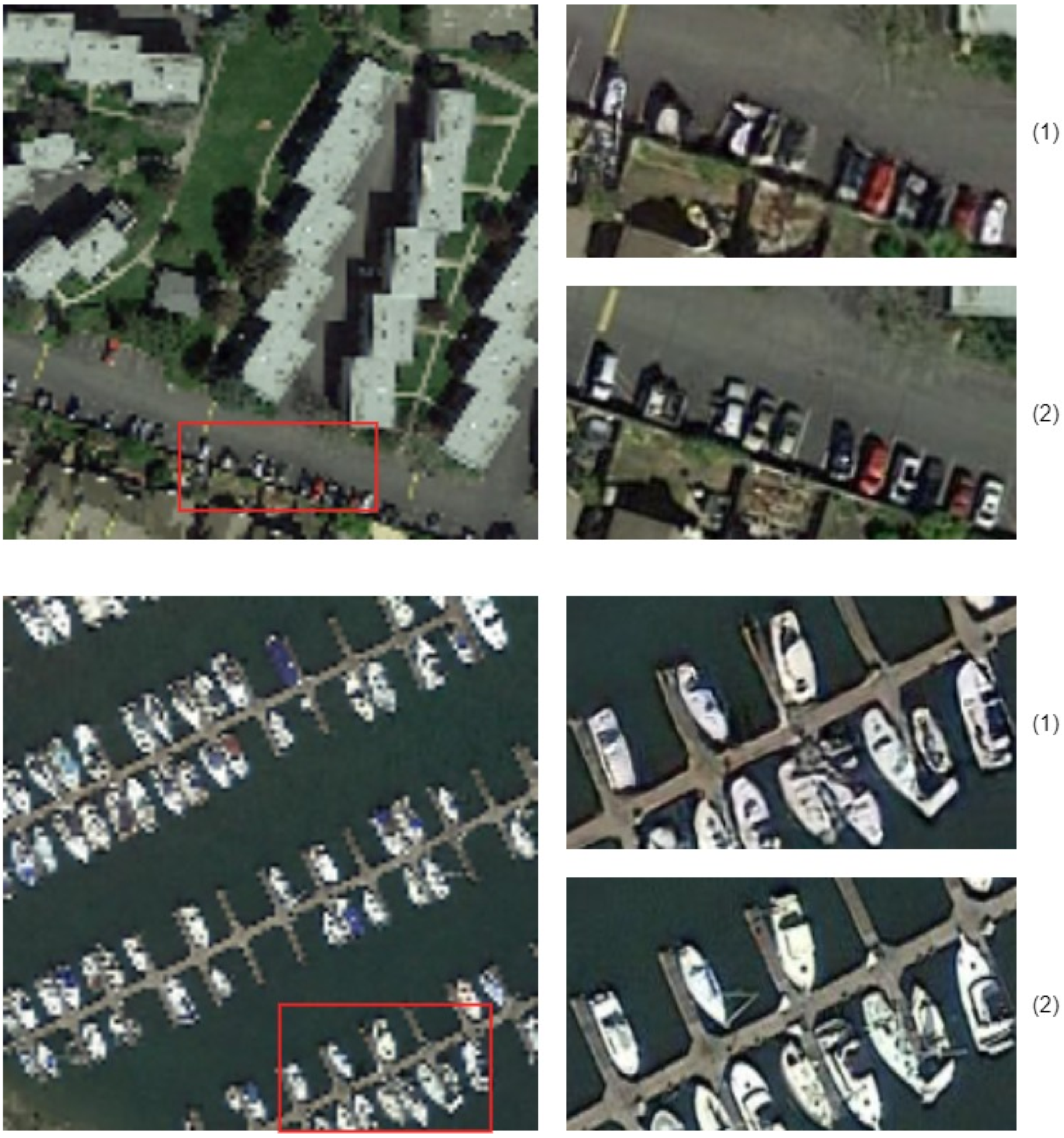

Figure 4.11: The red area shown on a low-resolution satellite image is zoomed to in the synthetic super resolution image (1) and the ground truth high-resolution image (2).

5 Discussion

In this chapter, the overall work done in this thesis is discussed with regards to the aim of exploring the usage of GANs in remote sensing. First, the results are analyzed and discussed in detail with relation to each research question split into sections similar to the method and results chapters. The method is then critiqued in approach, reputability, and reliability. Then, the work in a wider context is discussed.

5.1 Results

Generation

Experiment 1 evaluates the capabilities of generation and shows the ability to generate new synthetic data that is representative of the training data. When interpreting the results from DCGAN it is clear that the model is too simple to have real usage in remote sensing applications. Compared to real UAV images the distribution of generation is good, the model learned a diverse set of tree types to generate, but as aforementioned, the resolution is unusable for any real world application. Being just 64x64 pixels in size the quality of the images are fairly low and it is difficult to classify them as applicable as a data augmentation technique due to the relatively large data sizes of remotely sensed data. For example, in the case of augmentation of UAV images taken from a drone, the raw imagery is usually at least Full HD at 1920x1080 pixels and often times up to 4K resolution. DCGAN was introduced as one of the first extensions to GANs in 2015 and it can act as a good comparison metric to compare how much the models have improved in the past five years, but as a technique on its own it would not be applicable in remote sensing.

The low resolution results from DCGAN act as a starting point showing potential for the GAN's generation capabilities. By upgrading to a powerful model of StyleGAN2-ADA the results drastically improved. StyleGAN2-ADA shows how research within GANs has accelerated in the past five years. The generation resolution increased by over an order of magnitude from 64x64 up to 1024x1024 pixels. Even with such a jump in resolution, the quality of the images actually improved over DCGAN. The results are very sharp and represent accurate characteristics of the training data. The synthetic StyleGAN2-ADA images could be used in a machine learning pipeline as an additional data augmentation technique to generate synthetic data. As described in the results section, small amounts of mode collapse occur in most of the synthetic images. This could be due to the high resolution of the images or the lack of a larger training dataset. Training at a lower resolution might result in less mode collapse occurring due to smaller dimensionality in the generator as learning to generate fewer pixels is an easier task for the generator. As mentioned by the original styleGAN2 authors [21], the images are better described as sharpened 512x512 pixel images instead of true 1024x1024 images.

It is also important to note what the goal is when generating new data. It should not be to produce an exact replica of the training data, but rather to learn an approximation of the underlying characteristics that describe the data. While the generations might not be perfect and do contain visual artifacts, creating synthetic data that resembles something that could come from the same domain is feasible in most situations. The synthetic data might even get augmented even further which further lessens the importance of imperfections that exist in synthetic data. This is a fine line in data generation with GANs as when using small data sets the model can easily learn to simply memorize how the training data looks and not actually provide meaningful results. Sometimes the synthetic images look so real that they could convince someone that they are from the training set. This is difficult to inspect visually if the dataset contains more than a few hundred images. The FID is one method that tries to solve this problem. If the FID is 0 the generated data and the training data are identical. This gives an insight into how much a model might have overfitted a dataset but it is not perfect. If half the samples used in the calculation are replicas from the training set and half represent the dataset but are novel the FID would still give a score indicating good performance. Overall, the FID score gives good insight into how a model performs and is commonly used but accurately evaluating the exact performance of the models used in this experiment is difficult.

Image-to-Image Translation

The pix2pix model was trained to transform a segmented image into a real image. The results, in this case, are not ideal in translating between domains for all classes even though the training data contains a fairly even representation of buildings, forest, water, and ground. However, the struggle to generate a realistic building was hard due to the variances observable the in roof top colors and placement of chimneys or certain roof types. This made it difficult for the model to correctly generate a building as several roofs and buildings all were contained within one class. The gray class for ground is also problematic for the same reason as for buildings. Ideally each class in the segmentation mask should be clear and consistent with what it represents in the real image. Unfortunately, this dataset allows large variation within it's classes. The ground can contain roads, trails, green farmland, brown farmland, bushes, among other natural vegetation.

The results from this experiment showcase the importance of the training data in a machine learning model and show the potential for useable results in the case of conditional data augmentation for remote sensing applications. The provided method shows how to create synthetic data of satellite imagery and selecting the specific classes that are needed by simply creating images of the mask in the shape, arrangement, and quantity necessary. These images can then be used as input to the GAN in order to generate realistic looking imagery.

Cloud Removal

The imagery used for training CycleGAN in this experiment is mostly grassland covered areas with cities and water bodies during the summer season. This leads to images looking similar and being easier for the model to learn to remove the clouds and restructure the underlying satellite image. If this specific cloud removal GAN were to be tested in a new domain the performance would suffer. Therefore, this is not a generalized solution, but rather one that must be re-trained or fine tuned to apply in other domains where the satellite imagery might consist of a different landscapes or contain unseen features such as mountains.

Some artifacts also occur when an underrepresented class like water is present underneath a cloud which is to be removed. The model learns to interpolate that this could be grassland instead of water causing green spots in the water. This also conveys the difficulty of having one model for multiple domains.

Unfortunately, it is common that satellite imagery contain thick clouds along with thin clouds or haze. Thick clouds are impossible to to see through in visible light. The issue with training a machine learning model to remove thick clouds is the interpolation of the data underneath the clouds is near impossible with a single image. The complete loss of information makes it difficult to predict accurately what might exist underneath. A proven method is to use multiple images taken at different times. This is the best way to successfully remove clouds from satellite imagery by using temporal data in order to understand the environment that exists underneath the cloud cover and then interpolate and stitch together images that complete a cloud free image.

The cloud removal solution uses image-to-image translation and therefore it is possible to train a paired image model such as pix2pix. Since the data is simulated the access to paired training data is available and would potentially lead to better results, or at least faster convergence during training. However, CycleGAN was chosen in this task for the sake of evaluating a new model and to provide a broader exploration with unpaired training. A comparison of the two approaches would be an interesting experiment to test. As mentioned in the related work section, researchers [8] have tested a similar method to remove thin cloud cover, however this experiment using CycleGAN at 512x512 pixel resolution doubled the previous resolution and further proves the potential of the method.

Super Resolution

The results from training ESRGAN show that this model is capable of performing basic super resolution tasks in remote sensing. The small blob-like artifacts and small features blending into each other is a common occurrence with super resolution tasks. The information from a low resolution image makes it difficult to interpolate what data should lie in the extra pixels being added to the image. For example, if four pixels in a high resolution image contain two cars and they are down sampled, now these two cars may be represented by just one pixel. In the up sampling stage the model must generate what each of these four pixels should contain based off of one pixel.

In the samples presented in the results chapter it might be that enough training data containing similar objects and features were seen during the training phase of the GAN and therefore leading to more accurate super resolution results. The dataset used to train ESRGAN is commonly used for object detection tasks in satellite imagery. This made it a perfect dataset for super resolution as well because it includes many small objects that made learning how to up sample these objects easier. Without a good dataset, the results in the super resolution task might be considerably worse.

5.2 Method

The approach taken is one meant to be an exploration of the different ways that GANs can be applied within the domain of remote sensing. The method used consists of four distinct tasks that utilize GANs in an application area that is relevant to the domain of remote sensing. Since all the experiments used different GAN architectures and datasets the comparison between them is not done in this work.

The usage of a baseline when comparing machine learning models is necessary to have a comparison metric to appreciate how well a model performs. Unfortunately, with GANs it is an ongoing research question as to what the best way to compare GAN models is [4]. When generating synthetic images a quantitative metric is difficult to interpret. Therefore, several stages of visual inspection are necessary during training and when analyzing the results. This was done by generating images after every epoch and using a subjective decision if the quality is increasing or not. Of course the quantitative metrics that do exist give some insight to how close a feature extractor might think two images are by using metrics like FID that use inception weights in order to compare two distributions of fake and real images. It has been found that this score is not important in finding the best performing models with regards to generation. A feature extractor in a deep learning model will not be able to fully describe the quality of an image from a human perspective.

The qualitative assessment performed to determine the quality and usability of the synthetic images is a flawed one. The subjectivity of the domain will differ between every individual. A computer vision researcher will probably detect the mode collapse in experiment 1 and consider them poor generations, whereas a layman might consider them indistinguishable from the training data. In order to more accurately use a qualitative assessment ideally a large group of people should inspect many images. This could be done by conducting a sort of survey that randomly selected participants will determine the quality of the generation of imagery. This was not done in this thesis due to resource and time constraints but would have led to a more unbiased estimate of quality.

In experiment 2 and 3 regarding image-to-image translation a trained machine learning model to segmentate images could have been used to evaluate the performance. This process could be implemented by measuring the models performance on segmentation tasks in real samples and synthetic samples and then comparing the accuracy. This was not implemented due to being out of scope for this thesis but would have provided a better quantitative metric of generation quality without a human bias. A similar approach could also be applied within the super resolution task but swapping the segmentator with an object detector to detect the small features in both the synthetic and real satellite imagery.

5.3 The work in a wider context

The shift seen in the remote sensing community moving from classical methods of machine learning into deep learning over the past years is a key factor as to why this work is important. Currently, the usage of GANs in remote sensing is a new topic where much research has yet to be done. The work done in this thesis can impact this field by providing insight into how GANs can be used within remote sensing applications and to help grow further interest.

Looking beyond the field of remote sensing, the work done in this thesis regarding GANs can be applied to many other fields where computer vision is used such as medical imagery, surveillance, autonomous vehicles, and manufacturing. In all of these fields, the ability to generate synthetic data is a potential method to improve existing deep learning models. In the case of surveillance and autonomous vehicles, there will be times where imagery and video feeds are affected by haze or rain hindering a system's ability to detect features. A method similar to the one used in experiment 3 could be employed to train a GAN model for attempts to remove such distortions. These models relying on visual input also could benefit from super resolution demonstrated in experiment 4. Increasing the resolution of surveillance cameras, for example, to enhance the ability to detect dangerous items or situations is one such use case.

While the positivity and excitement behind generative models such as GANs are growing, the act of generating synthetic data can also be used in a malicious manner. An article posted on The Verge[1] describes research done to create synthetic remote sensing data, the same work is referenced in this thesis in the related work section. The article goes on to describe the malicious implications of generating fake data and specifically a worry about fake satellite imagery. GAN generated satellite imagery could fool people into believing there is a flood or wildfire, or be used to mislead rival countries by spoofing military planning sites. Even though the vast majority of work done in machine learning, and with GANs specifically, are positive and provide beneficial usage, the malicious and ethical use of machine learning should always be considered.

[1] https://www.theverge.com/2021/4/27/22403741/deepfake-geography-satellite-imagery-ai-generated-fakes-threat

6 Conclusion

This chapter begins by answering the research questions this thesis proposed. Then, a collection of suggestions regarding future work that can be done beyond this work is presented.

6.1 Research Questions

The aim of this thesis was to investigate and explore some of the applications of GANs in the domain of remote sensing. These applications include the generation of high-resolution aerial imagery from UAVs, image-to-image style transfer between a segmentation mask and real imagery, removal of thin clouds from satellite imagery, and super resolution of satellite imagery. The four experiments set out to answer the proposed research questions. The first two experiments prove how GANs can create synthetic remote sensing data. This data is useful for a variety of remote sensing applications such as training a deep learning model for detecting objects in UAV or satellite images or segmentation for land use classification. The third experiment showed how a GAN can remove thin cloud cover which is a common task encountered by anyone using satellite imagery. The fourth showed how a GAN can perform a super resolution task. Higher resolution data is usually more desirable, yet more expensive, in remote sensing tasks. This makes this process of enhancing the resolution of satellite imagery using a GAN useful. The research questions are summarized below:

1. At what resolution and quality can synthetic aerial forest imagery be generated with GANs?

   The synthetic images generated by GANs are impressive and using state of the art models such as styleGAN2-ADA become almost indistinguishable from the real images generating samples at a resolution of 1024x1024 pixels.

2. Can a GAN learn the mapping between segmented satellite masks and real satellite imagery using paired image-to-image translation?

   Yes, the possibility to train a GAN is shown despite the issues encountered with data distribution. The data directly controls the results as is the case with most machine learning applications. Given a specific domain where there was a need for conditional data augmentation the pix2pix model is a plausible solution for generating specific synthetic data.

3. Can a GAN learn to remove thin cloud cover from satellite imagery using unpaired image-to-image translation techniques?

   Yes, CycleGAN can learn to remove thin cloud cover from satellite imagery. The results are overall great except for the few cases containing the less represented class which was similar to what was found in experiment 2. The synthetic imagery is high quality, and the majority of test samples successfully remove all thin cloud coverage in the image.

4. Can a GAN be used in a super resolution task using satellite imagery to generate higher resolution satellite imagery?

   Yes, ESRGAN shows promising results in super resolution with satellite imagery leaving almost no major artifacts. These results would also be applicable using any remote sensing data such as UAV imagery where higher resolution is also desirable.

6.2 Future Work

There are many possibilities to expand on the work done in this thesis. The research of GANs has grown quickly and interest in generative models continues to grow. Due to the rapidly moving research there will most likely be new state of the art models in the domains presented in this thesis within months. Therefore, evaluating these new architectures and changes made by researchers again in similar experiments would be an interesting evaluation. It could be the case that in certain applications within remote sensing some model outperforms another based on what it specializes to do.

Throughout the experiments in this thesis only remotely sensed imagery in the RGB spectrum was used. Beyond the visible light spectrum there are many possibilities to use GANs in new and exciting ways. A GAN could learn the mapping between a multi-spectral image in infrared or radar and in RGB or vice versa. This can be useful in cases where certain remote sensing data is only captured in one of the spectrum and the other is desired. The usage of multi-spectral imagery for tasks such as cloud removal using GANs could also be further researched with the help of remotely sensed radio imagery. The radio imagery has the ability to penetrate cloud cover and build a physical model of what lies underneath. In combination with a GAN it could be possible to use the physical model and use image-to-image translation techniques to convert this model into a realistic representation of the surface below the clouds.

As mentioned many times throughout this thesis, the evaluation of GANs is a difficult task. In addition to evaluating one GAN, the comparison between different GAN models is an ongoing research topic as well. A suggestion of work could be one regarding the best way to compare synthetic remote sensing data generated by GANs. There is research done on comparison as a general evaluation, but within remote sensing nuances due to small features in satellite imagery and very high-resolution from UAV images might lead to new insights. For example, a standard could be set that an object detection model must be able to perform at a certain accuracy for the synthetic data to be classified as usable.

Finally, to achieve a deeper understanding of one of the applications introduced in this thesis a more in-depth evaluation of any of the experiments could be done. The work done in this thesis is a broad exploration and focusing on one specific use case would allow for a clearer picture of the overall usability and applicability of GANs in that setting. Overall, much research is still necessary to get a better understanding of the usability of GANs in remote sensing.

A Appendix

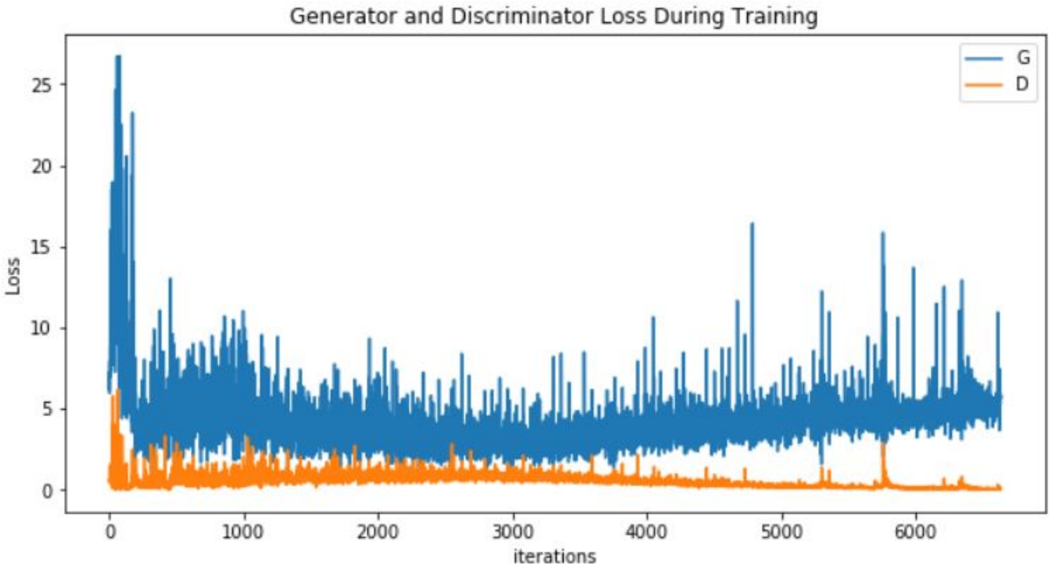

Figure A.1: DCGAN Loss Plot

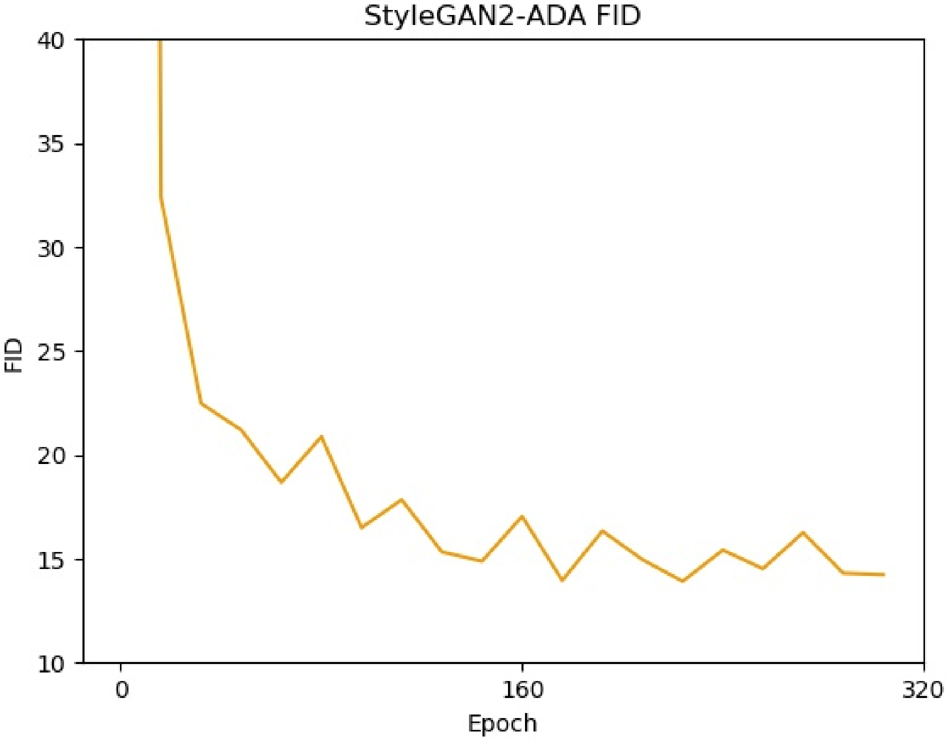

Figure A.2: StyleGAN2-ADA FID Values

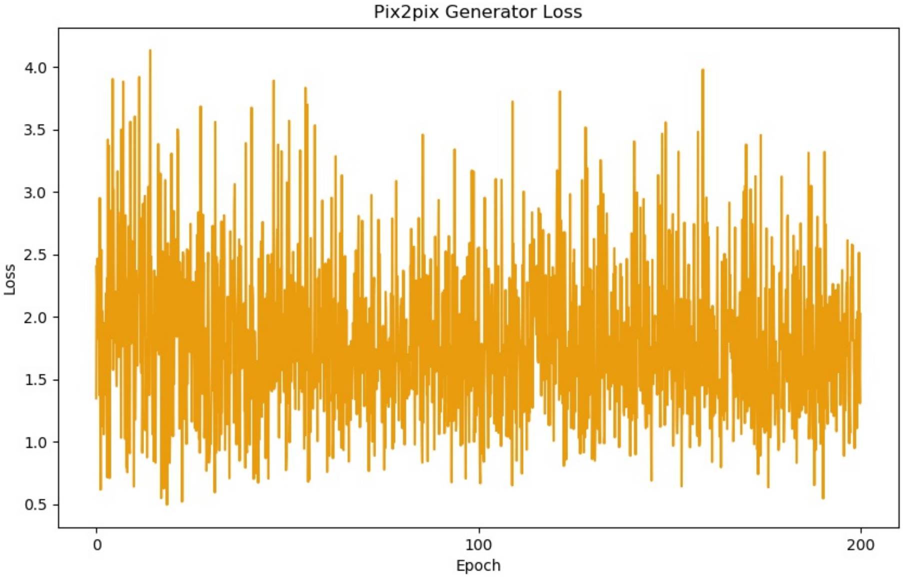

Figure A.3: Pix2pix Generator Loss Plot

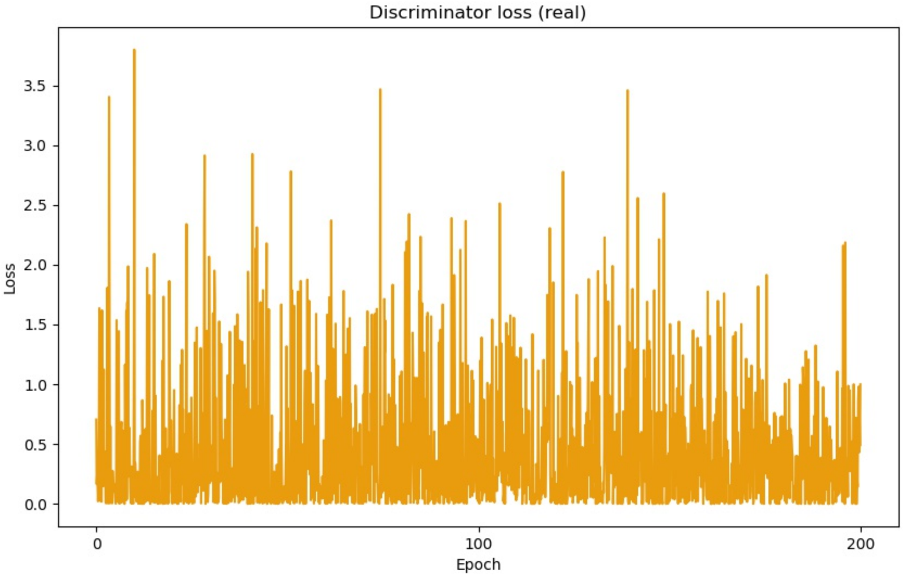

Figure A.4: Pix2pix Discriminator Loss for real samples

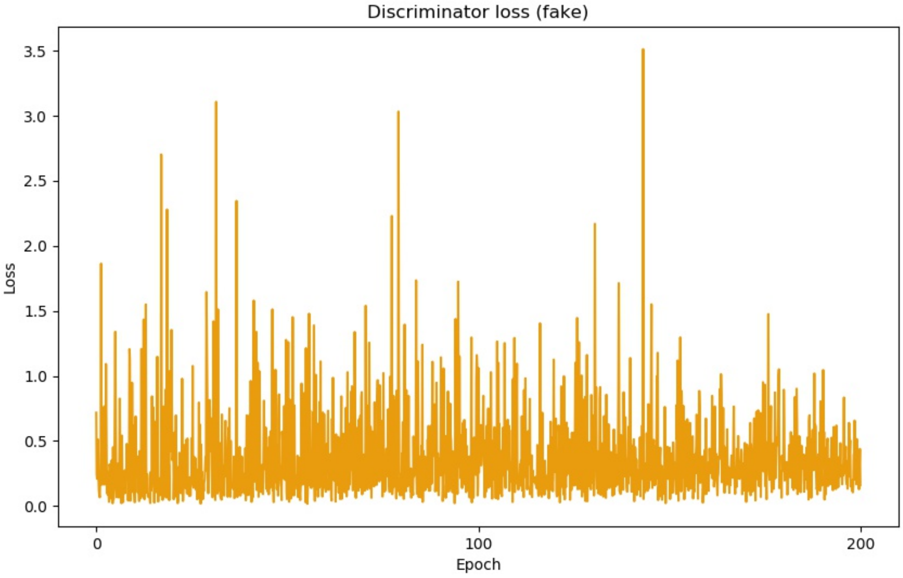

Figure A.5: Pix2pix Discriminator Loss for fake samples

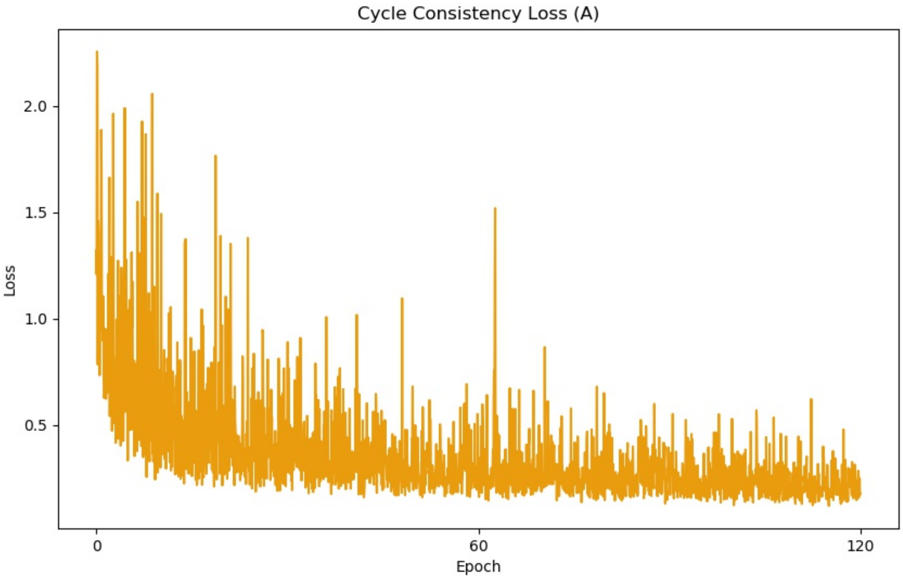

Figure A.6: CycleGAN Cyclic Loss for Domain A

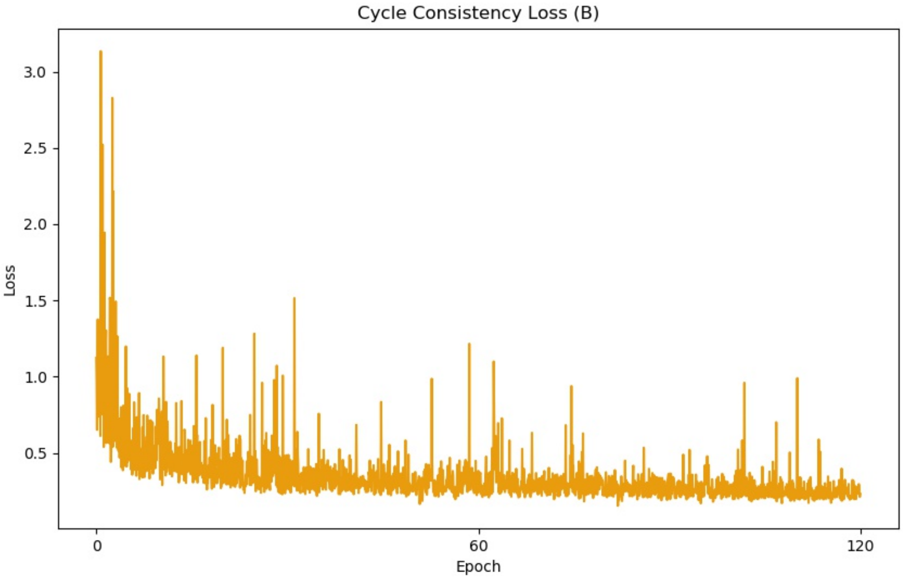

Figure A.7: CycleGAN Cyclic Loss for Domain B

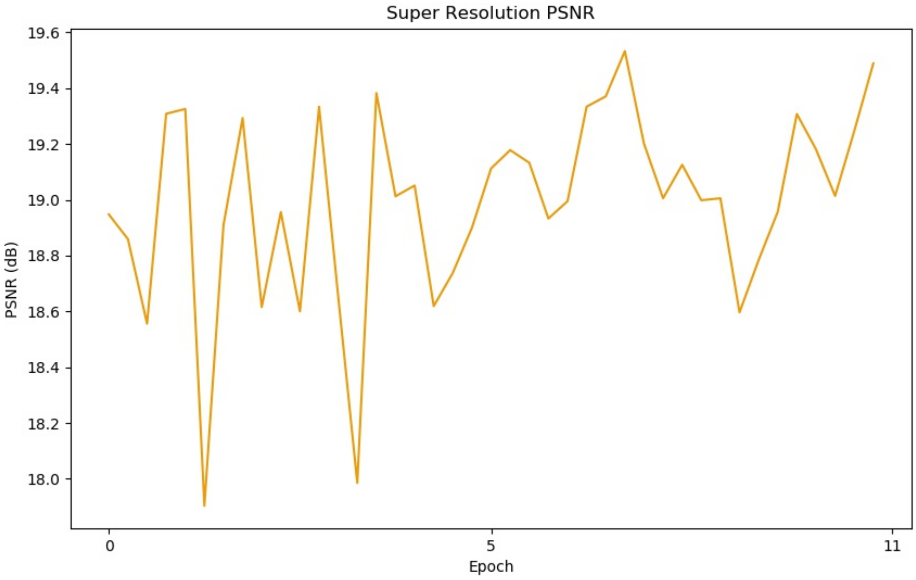

Figure A.8: ESRGAN PSNR Values

Bibliography

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. *Data Augmentation Generative Adversarial Networks*. 2018. arXiv: 1711.04340 [stat.ML].

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

[3] Mariana Belgiu and Lucian Drăguţ. "Random forest in remote sensing: A review of applications and future directions". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 114 (2016), pp. 24–31. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2016.01.011. URL: https://www.sciencedirect.com/science/article/pii/S0924271616000265.

[4] Ali Borji. "Pros and cons of gan evaluation measures". In: *Computer Vision and Image Understanding* 179 (2019), pp. 41–65.

[5] James B Campbell and Randolph H Wynne. *Introduction to remote sensing*. Guilford Press, 2011.

[6] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. "Image super-resolution using deep convolutional networks". In: *IEEE transactions on pattern analysis and machine intelligence* 38.2 (2015), pp. 295–307.

[7] W. Dong, X. Zhang, and C. Zhang. "Generation of Cloud Image Based on Perlin Noise". In: *2010 International Conference on Multimedia Communications*. 2010, pp. 61–63. DOI: 10.1109/MEDIACOM.2010.77.

[8] Jianhao Gao, Qiangqiang Yuan, Jie Li, Hai Zhang, and Xin Su. "Cloud Removal with Fusion of High Resolution Optical and SAR Images Using Generative Adversarial Networks". In: *Remote Sensing* 12.1 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12010191. URL: https://www.mdpi.com/2072-4292/12/1/191.

[9] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. "Image style transfer using convolutional neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2414–2423.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[11] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML].

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *arXiv preprint arXiv:1706.08500* (2017).

[13] Xun Huang and Serge Belongie. "Arbitrary style transfer in real-time with adaptive instance normalization". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1501–1510.

[14] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 448–456. URL: `http://proceedings.mlr.press/v37/ioffe15.html`.

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-To-Image Translation With Conditional Adversarial Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. 2016. arXiv: `1603.08155 [cs.CV]`.

[17] Alexia Jolicoeur-Martineau. "The relativistic discriminator: a key element missing from standard GAN". In: *arXiv preprint arXiv:1807.00734* (2018).

[18] A Jordan et al. "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes". In: *Advances in neural information processing systems* 14.2002 (2002), p. 841.

[19] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of gans for improved quality, stability, and variation". In: *arXiv preprint arXiv:1710.10196* (2017).

[20] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. *Training Generative Adversarial Networks with Limited Data*. 2020. arXiv: `2006.06676 [cs.CV]`.

[21] Tero Karras, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4401–4410.

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. "Analyzing and improving the image quality of stylegan". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 8110–8119.

[23] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[25] Jakub Langr and Vladimir Bok. *GANs in action: deep learning with generative adversarial networks*. Manning, 2019.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[27] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST handwritten digit database". In: *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[28] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690.

[29] *Loss Functions nbsp;| nbsp; Generative Adversarial Networks nbsp;| nbsp; Google Developers*. URL: https://developers.google.com/machine-learning/gan/loss.

[30] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. "Are gans created equal? a large-scale study". In: *arXiv preprint arXiv:1711.10337* (2017).

[31] Lei Ma, Yu Liu, Xueliang Zhang, Yuanxin Ye, Gaofei Yin, and Brian Alan Johnson. "Deep learning in remote sensing applications: A meta-analysis and review". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 152 (2019), pp. 166–177. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2019.04.015. URL: https://www.sciencedirect.com/science/article/pii/S0924271619301108.

[32] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[33] Giorgos Mountrakis, Jungho Im, and Caesar Ogole. "Support vector machines in remote sensing: A review". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.3 (2011), pp. 247–259. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2010.11.001. URL: https://www.sciencedirect.com/science/article/pii/S0924271610001140.

[34] John Nash. "Non-cooperative games". In: *Annals of mathematics* (1951), pp. 286–295.

[35] Roope Näsi, Eija Honkavaara, Päivi Lyytikäinen-Saarenmaa, Minna Blomqvist, Paula Litkey, Teemu Hakala, Niko Viljanen, Tuula Kantola, Topi Tanhuanpää, and Markus Holopainen. "Using UAV-based photogrammetry and hyperspectral imaging for mapping bark beetle damage at tree-level". In: *Remote Sensing* 7.11 (2015), pp. 15467–15493.

[36] Mohammad Pashaei, Michael J Starek, Hamid Kamangir, and Jacob Berryhill. "Deep Learning-Based Single Image Super-Resolution: An Investigation for Dense Scene Reconstruction with UAS Photogrammetry". In: *Remote Sensing* 12.11 (2020), p. 1757.

[37] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[39] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. *Improved Techniques for Training GANs*. 2016. arXiv: 1606.03498 [cs.LG].

[40] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of Big Data* 6.1 (2019), pp. 1–48.

[41] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[42] Praveer Singh and Nikos Komodakis. "Cloud-gan: Cloud removal for sentinel-2 imagery using a cyclic consistent generative adversarial networks". In: *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2018, pp. 1772–1775.

[43] Joshua Susskind, Adam Anderson, and Geoffrey E Hinton. *The Toronto face dataset*. Tech. rep. Technical Report UTML TR 2010-001, U. Toronto, 2010.

[44] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks*. 2018. arXiv: 1809.00219 [cs.CV].

[45] Q. Wu, Y. Chen, and J. Meng. "DCGAN-Based Data Augmentation for Tomato Leaf Disease Identification". In: *IEEE Access* 8 (2020), pp. 98716–98728. DOI: 10.1109/ACCESS.2020.2997001.

[46] Chunxue Xu and Bo Zhao. “Satellite Image Spoofing: Creating Remote Sensing Dataset with Generative Adversarial Networks (Short Paper)”. In: *10th International Conference on Geographic Information Science (GIScience 2018)*. Ed. by Stephan Winter, Amy Griffin, and Monika Sester. Vol. 114. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 67:1–67:6. ISBN: 978-3-95977-083-5. DOI: `10.4230/LIPIcs.GISCIENCE.2018.67`. URL: `http://drops.dagstuhl.de/opus/volltexte/2018/9395`.

[47] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.