Universiteit Utrecht

Faculty of Science

Master Thesis

Artificial Enhancement of Remote Sensing Data using Generative Adversarial Networks

Author:
Bryan Weiss

Primary Examiner:
Dr. Z. Yumak
Secondary Examiner:
Dr. A.J. Feelders
Supervisor:
O. Fabius

March 2, 2020

Universiteit Utrecht

1 Acknowledgements

I would first like to thank Zerrin Yumak and Ad Feelders for their time and consideration in acting in the role of examiners for this thesis. I would like to additionally thank Zerrin for the help and time she's given towards supporting this project, especially through the various complications that arose.

I also want to thank Otto Fabius as well as Sobolt for their assistance and resources without which I would not have been able to conduct this research.

Lastly, I would like to thank Alexander Los, for his help and the expertise he provided in complex tropospheric data. Much of the progress made in this work was thanks to his knowledge.

2 Abstract

Remote sensing is the process of obtaining informative data about an object from afar. While this applies to many different methods of data collection, as well as different domains to collect data from, there is a universal constant that the technology and recording instruments utilized towards this purpose are being improved day by day. However, for every piece of higher quality data collected from these new instruments, there still exists much more data from now lower quality measurement instruments that can not provide the historic significance nor insights that will begin to be made with increases in accuracy.

Machine learning has shown the capability to recognize very subtle patterns between different types of data. In recent years, one such method known as Generative Adversarial Networks (GANs) has displayed much success in artificially creating new data based on given input by learning from corresponding example output. Through this research, we show the potential for using the complex generative abilities of GANs to improve the accuracy and quality of remote sensing data taken from older instruments by using more precise data from newer technology as examples to learn from.

We take data obtained from two atmospheric satellites utilizing two Ozone measurement instruments TROPOMI and its predecessor OMI that collect Nitrogen Dioxide (NO_2) readings in the troposphere, early indicators of pollution, and use it to create paired datasets based on location and the time a location was crossed over by each satellite. Using this training set, we train an Enhanced Super Resolution GAN (ESRGAN) to improve both the resolution and measured values of OMI data used as input, inspired by TROPOMI training examples.

Contents

3 Introduction

The ability to process and analyze data is a trait both unique to, and highly sought after by humans as doing so provides us a better understanding of that data, and by extension, our environment. This understanding inevitably gives us greater freedom and flexibility to act upon that environment and change it to better suit our needs. Therefore, it goes without saying that methods to better analyze or improve data are just as important. Of the different environments that people are interested in understanding, few may be of as large a scope as the Earth itself. Remote sensing is often being used to collect valuable data about the Earth with the use of expensive equipment such as high flying air crafts or satellites. This paper aims to provide the basis for a new method of artificial data enhancement for remote sensing data. It will focus on the collection and improvement of complex remote sensing data, primarily nitrogen dioxide readings in the troposphere.

3.1 Background

One method for processing complex data that has seen marked success in recent years is known as machine learning. Machine learning involves making use of a computer's superior processing speed and memory to better perform complex algorithms without explicit instructions [22]. It includes algorithms necessary to study, and even improve upon data. One such method involves passing data through a layer of nodes that each add a weight to the input, recognizing and remembering patterns in the data. This framework of modeling, called an artificial neural network, is named after the collection of neurons that analyze information in biological brains [13]. Taking this concept even further, if more layers are added to the neural network creating a deep neural net, the resulting algorithm is referred to as deep learning. Deep learning has thus far shown tremendous promise in accurately analyzing large sums of data reaching conclusions that people cannot imagine. Using deep learning, neural networks can even learn to produce data rather than just analyze it. One such application of this is known as super resolution, that is, improving the quality of low resolution images through predictions of pixel values made by the neural network [29]. One type of implementation in general has seen rapid support in recent years for its ability to produce new and high quality of data. Generative Adversarial Networks, or GANs, work by pitting two separate neural networks against each other as adversaries in a zero sum game [5]. One network, called the generator, attempts to produce data that will successfully fool the other, while its opponent, the discriminator, attempts to differentiate fake data from real. As both networks become successively better at outwitting each other, eventually the fake data becomes more or less indistinguishable from the real.

Currently, two satellites equipped with gas measurement tools called TROPOMI and its predecessor, OMI, are measuring the Earth's air quality in close succession. NASA is planning to move the OMI satellite to a shifted orbit to measure the air quality at a different time. By using the aforementioned techniques to produce data, it should also be possible to perform super resolution, or forcibly improving the resolution between two image pairs of lower and higher resolution respectively, to enhance the data quality of the OMI satellite to that of the TROPOMI. To effectively gain data equal to that of two TROPOMIs without building and launching an extra satellite would greatly increase the data gathering capabilities at reduced cost. As there are many other factors in play than simple image resolution in the data containing air quality, this isn't truly a super resolution task. Not only is the data being enhanced not of images, but of highly precise gas readings, but also aren't exact pairs due to time passed between readings and weather conditions. To counteract these limitations, our method will rely on the more perceptually advanced learning method of GANs to overcome it. The question remains whether it will be possible to enhance the data acquired from OMI using the highly adaptive and replicative nature of GANs to an acceptable degree.

While GANs have shown great potential in the field of the reproduction and improvement of data, there are still some problems they face which can prevent successful application. The most commonly occurring issue perhaps, is referred to as mode collapse, where the generator learns to produce a few successful examples of data and so will not generate any others. On the opposing side, it's possible for the discriminator to become too successful at discerning real data from fake resulting in a reduced generator gradient preventing the generator from further learning. What may be the most important issue, however, is that of non-convergence [5]. The ultimate goal of a GAN's zero sum game, is reaching a converging point where the discriminator can no longer tell the difference between real and fake data. In practice this is very difficult to achieve as the results instead tend to oscillate, and destabilize, never reaching convergence. These issues will be explained in more detail in section 4.6.3. Some implementations of GANs have focused on reducing the occurence of these issues in order to achieve an overall better performance. For instance, one specific implementation, known as CycleGAN, involves a cyclic movement of data such that what is generated as output, can also be fed back into the system in reverse to test if it indeed produces the same input to see if mode collapse has occurred and train to lessen future occurences [1]. Through this research, we seek to answer whether the accuracy and quality of OMI-collected satellite data can be improved to that of TROPOMI using GAN related techniques.

3.2 Research Questions

In summary, the following research questions can be clearly defined and pursued through the rest of this paper.

> **1.** *Can GANs be leveraged to improve complex data relying mainly on perceptual learning?*
> **2.** *Can data collected from the low resolution OMI satellite be improved to the resolution of TROPOMI using GANs data enhancement techniques?*

3. *Will the accuracy of OMI data readings suitably improve when enhanced to TROPOMI level resolution?*

3.3 Motivation

With the increasing issue of air pollution and the effects it has on our environment, it becomes ever more necessary to monitor and track the changes and consequences to air quality. The ability to improve the results of our older technology has wide reaching applications not only in allowing us to obtain more frequent and higher quality air monitoring with the current two satellites, but also in having the means to further improve this data in the future as well. This not only concerns future data, but also past data. The OMI satellite currently has over 12 years of historical data that could stand to be improved to the level of TROPOMI, allowing many more trends and statistics to be made which weren't possible before. GANs show a remarkable ability to generate data compared to earlier attempts at generative machine learning algorithms. This in turn makes them a suitable method for analyzing and producing remote sensing data which can also be made up of a large amount of complexity that can thus be difficult to successfully derive patterns from. A successful prototype applied to the improvement of NO_2 data in the troposhpere could be extended to also improve methane, or C02 data among others. Following this reasoning, data improvement of obsolete models to match newer advancements could be utilized in an even wider variety of domains not limited to remote sensing.

3.4 Contributions

Through this work, contributions have been made in the form of methods for both collecting and enhancing complex NetCDF NO_2 tropospheric data. More specifically, we have created a process that takes very low level NO_2 data with high difficulty of readability and reformats it into easily displayable and pairable datasets for both OMI and TROPOMI by location and time. While this contribution is significant in itself, we have also modified and trained a variant of Xintao's Enhanced Super Resolution GAN (ESRGAN) [27] on this paired data to accept and then produce OMI and generated TROPOMI swaths of data respectively displayable over a map of the Earth. These contributions may be useful on their own for researchers and analysts of tropospheric data or by providing the basis of an application that could enhance and measure OMI tropospheric data.

3.5 Outline

In section 4, we'll examine the various literature explaining more in depth about the relevant data, methods of enhancement, GANs, and other concepts presented in this thesis. Section 5 will cover details behind popular and successful GANs as perspective candidates for the main tasks that will be explored. Section 6 will give a brief overview of other projects with similar aims to this one in using GANs to act on and improve remote sensing data. Section 7 gives a more detailed account of the data used for this task in addition to how it was collected and limitations associated with it. In section 8, we'll explain the methods we used to create the datasets used for training, as well as setting up the training process to produce the results. Finally the results will be overviewed in section 9 followed by the conclusion to this paper in section 10.

4 Literature

In this section, various technologies and terms relevant to understanding the proposed concepts and research focus will be explained. It will begin with brief explanations of the data and where it comes from, followed by what machine learning is, and specifically neural networks. The roots of the main task involving super resolution will also be detailed. Finally, the primary focus of this paper, pertaining to GANs, will be described in more detail.

4.1 Remote Sensing

Remote sensing, from the broadest definition, is being able to obtain information about an object from a distance without touching it. This applies to not just physical objects like land or water surfaces, but also various kinds of energy, temperature, gasses, or the atmosphere to list a few examples [33]. Many kinds of measurement tools have been developed specifically for this purpose, and have also been attached to far ranging vehicles such as aircrafts, ships, or even satellites orbiting the Earth from space. Gasses in the Earth's atmosphere can be measured by satellite instruments recording the UV backscatter as sunlight hits these gasses in the troposphere and stratosphere [36].

4.2 Troposphere

The troposphere is the first layer in the Earth's atmosphere, and also where most weather events occur. As it is closest layer to the surface, it functions as an early indicator of greenhouse gasses released into the air. Several gasses can be measured in this layer of the atmosphere including nitrogen dioxide (NO2), carbon monoxide (CO), and ozone. There are several reasons we have chosen to use NO2 measurements for this work. Both CO and ozone would also be useful gasses to improve the accuracy of current measurements, however, CO is not among the available measurements taken by the OMI instrument and ozone has too short of a lifetime to reliably compare between OMI and TROPOMI.

NO2 on the other hand is easily detectable and measureable and has had a long history of being measured from space. It's well understood what chemical reactions result from NO2 as well as what causes it to appear in the atmosphere. NO2 is also a precursor for many other gasses that are caused by pollution [36].

4.3 Machine Learning

Machine learning is a method that uses the significant processing and computation power that computers now possess to formulate complex algorithms to process data [22]. Given large amounts of relatable data, a machine can draw inferences from that data and recognize minute patterns that a human might otherwise be unable to observe. Using the machine's larger memory for bytes of data, these patterns can continuously be compared with more and more data and slowly refined giving the algorithms more certainty. This is the learning phase for the machine as it must be given copious amounts of data in order to successfully draw useful inferences. Two issues that can occur with machine learning, however, is that of underfitting and overfitting the data. In the case of underfitting, the complexity of the model might be too low that it is unable to meaninfully distinguish between different kinds of data. This prevents the model from learning anything useful. On the opposite side, overfitting occurs when the parameters given to the model matches the data too closely such that it becomes biased to the observed data and can not gain any useful information from new data that is given [22]. In order to prevent underfitting and overfitting, the model should be complex enough that conclusions can be drawn from the data, as well as having a large enough amount of data that a bias isn't formed.

4.4 Neural Networks

Artificial Neural Networks (ANNs) were devised based on the biological representation of the brains of animals. Basically a weighted directed graph is created of nodes representative of artificial neurons that are connected by directed edges containing weights [13]. The graph is made up of several layers including an input and an output layer along with one or more hidden layers between them. This structure forms a connection between the inputs of each neuron and their ouputs. A general example of an ANN is shown in Figure 1. Input flows through the different nodes of the input layer, modified by the the different weights of each node in the hidden layer before finally producing output from the output layer.

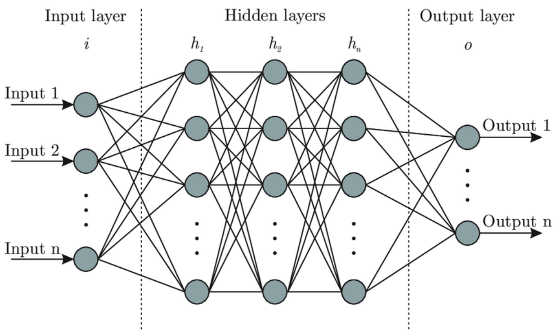

Figure 1: Artificial Neural Network Architecture [31]

There are two types of architectures that are commonly seen in ANNs, feed-forward and feedback networks. In feedback networks, connections are made between nodes in both directions, allowing for loops to occur. They are in this way dynamic systems that can modify the neurons when new input appears to reach new network states. Unlike feedback, feed-forward networks lack any sort of looping and are only uni-directed graphs. These networks are static and produce a single set of output values for each input [13]. The most commonly used type of feed-forward network is the multilayer perceptron. Multilayer perceptrons are networks that contain multiple hidden layers in addition to the input and output layers, that are feed-forward connected without any connections between nodes of the same layer. The popularity of this structure stems from the development of the back-propagation learning algorithm that employs gradient descent to determine the weights of each node in the perceptron [13]. Back-propagation works by computing the error observed in the output layer of the network, and then propagating those values backwards through the network updating the weights of each node with the new corresponding deltas. Effectively this method allows the network to eventually learn the weights needed to produce accurate output. This learning can either be carried out in a supervised or unspervised manner. Supervised learning involves using data for which a correct answer can be determined from the output of the network. Using those answers as a guide, the network can then learn to improve and adjust. Unsupervised learning in contrast forgoes this student-teacher relationship. It tries to find patterns just within the different inputs of the dataset to understand and organize those patterns into categories [13]. Although both methods of learning have specific benefits, GANs generally make use of supervised learning multilayer perceptrons in order to teach the generative model from back-propagated error.

4.4.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a special instantiation of ANNs. They're set up as a series of stages such that inputs pass through convolutional layers, followed by pooling layers, followed by non-linearities.

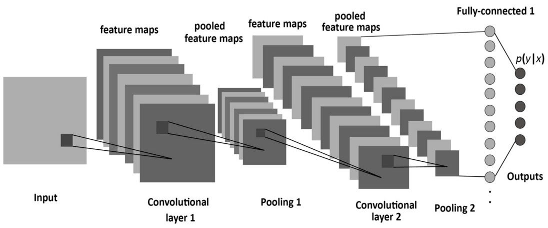

Figure 2: Convolutional Neural Networks [32]

The convolutional layer is made to detect features present in local conjunctions, while the pooling layer merges semantically similar features [24]. As shown in Figure 2, a filter passes over the input data which splits that data into multiple feature maps. Those feature maps are then pooled together which can then be split up into more feature maps with additional convolutional layers and then pooled again. After the defining and merging features in each of the layers, the network can perform back-propagation to update and train all of the weights like a standard ANN. For these reasons, CNNs have been very useful in networks that perform object detection as well as feature recognition. GANs also benefit greatly from CNNs as feature detection is extremely useful in data generation.

4.5 Super Resolution

The concept of super resolution stems from the wish to improve the innate resolution of viewed images. The ability to improve data quality, as mentioned earlier, is highly sought after in data analytics. This is especially true when concerning satellite and aerial imagery which the resolution of is not always good enough to ascertain distinguishable features from. Normally, when an image is upscaled from a low resolution to high, it inevitably will suffer from blurring, as it must create new pixels to fill in the gaps between the originals [29]. Several methods exist for trying to do this already with standard image transformation such as pixel replication or cubic spline interpolation do little to avoid these problems [29]. Due to the fact that naive pixel interpolation wasn't enough to improve resolution data, super resolution came about as a method using learning-based networks to predict the missing data needed to transition from lower to higher resolutions of images using image generation.

4.6 GANs

The GAN framework, proposed by Goodfellow et al. in 2014, has seen much greater success compared to other existing deep generative models [5]. As mentioned earlier, the idea is that two networks, a generator and a discriminator, compete in a zero sum, minimax, game where each network tries to outdo the other:

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

Using two multilayer perceptron networks, D and G, a value function V(D, G) is used to calculate the log liklihood between discriminated real data D(x) and discriminated fake data D(G(z)). D tries to maximize this probability, whereas G tries to minimize it [5]. This calculation becomes the adversarial loss with which the generator and discriminator use to adjust the weights of each node for a better result. You can picture a counterfeiter(G) and a police officer(D) working to produce and detect fake currency respectively. As each network sequentially learns from the other about the difference between real and fake, both networks improve upon their own tasks until real becomes indistinguishable from fake. In this way, the system tries to reach a nash equilibrium, where each network can do no better than the other [5]. The basic architecture of a GAN can be viewed in Figure 3. The figure depicts two generators, D and G, with paired data z and x being fed into the generator and discriminator rexpectively. The generated data G(z) is compared the real data x within the discriminator which produces pairs of real and fake data that both networks can learn from.

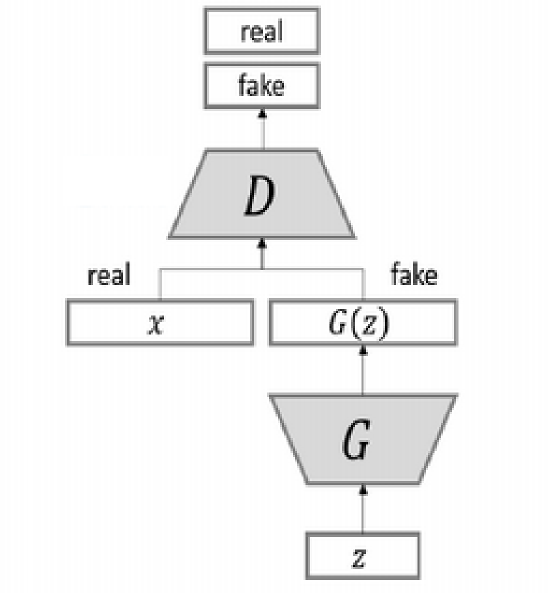

Figure 3: GAN architecture [14]

4.6.1 DCGAN

Since GANs rely on ANNs by their intrinsic nature, the complexity and efficiency of those networks are extremely important to the efficacy of the GAN itself. As deep convolutional networks have proven to be very effective learning architectures for conventional ANNs, it goes without saying that these methods would be desirable in GANs as well. Deep Convolutional GANs (DCGANs), as could be guessed, are GANs utilizing deep convolutional networks in their implementation, which historically, had met with little original success [21]. Three key points differentiate the first successful DCGAN from previous attempts. First, they replaced all pooling layers with strided convolutions so that the networks would be able to learn spatial upsampling. They also normalized the batch data being presented to the generator and discriminator to have zero mean and unit variance to better stabilize learning. They also removed any fully connected hidden layers to further improve stability at the cost of convergence speed [21]. Due to their improvements in stability and accuracy, DCGANs have been used as the core structure for many more improvements in GAN training.

4.6.2 SRGAN

SRGAN, or Super Resolution GAN, as the name suggests, is a GAN implementation focused towards performing super resolution. SRGAN replaces the usual mean squared error content loss a feature map loss that is more invariant to changes in pixel space [2]. This is done by optimizing each block of the model using the perceptual loss gained from the GAN framework. This change is helpful for improving image resolution because the error values between the pixels won't end up changing that much when increasing resolution, however the space each pixel takes up will inevitably increase and is more beneficial for the generator to learn from. SRGAN generally takes input images that are one fourth the resolution of the high resolution output and real images that they're paired against [2].

4.6.3 Issues

Despite the goal of the minimax game played by GANs is to reach an equilibrium, this is actually very difficult in practice. In fact, the main issue that GANs face is that of non-convergence [15]. Balancing the learning of the discriminator and generator can be like walking a tight-rope. If the learning rate of the generator is too large, it can continuously overshoot the optimal solution without ever reaching the point of convergence with the discriminator. On the other hand, if the discriminator learns much more quickly than the generator, then the system could run into the vanishing gradient problem, where the discriminator starts rejecting the data created by the generator with high confidence, causing the generator's gradient to diminish, and subsequently, no longer improve [15].

Another issue that can cause trouble when attempting to train a GAN is that of mode collapse. It occurs when a generator ends up mapping multiple inputs to the same output, limiting the amount of samples that can be produced from real data. This can happen when the generator learns few successful occurrences that are able to fool the discriminator and so draws the wrong conclusions from that [15]. According to a proof provided by Barnett(2018), generators are actually forced towards mode collapse unless the discriminator has already been trained to optimality [15]. This, however, directly contradicts our understanding that the generator and discriminator must learn at an equal pace.

5 Known Methodologies

In order to create the seemingly paradox nature of a stable GAN, many techniques and strategies have been created to varying success. Although these different implementations all seek to stabilize the process of training GANs, they each present very different ways of doing so. In this section we will detail some different methods that have enjoyed increased popularity in recent years.

5.1 CycleGAN

A further implementation of the GAN framework, called CycleGAN, involves the addition of an extra pair of generator and discriminator networks. Like the original GAN, CycleGAN defines a loss function based on the minimax game between each pair of networks:

$$\begin{aligned}\mathcal{L}_{\text{GAN}}\left(G, D_z, z, x\right) = & \ \mathbb{E}_{x \sim p_{\text{ data }}(x)}\left[\log D_z(x)\right] \\ & + \mathbb{E}_{z \sim p_{\text{ data }}(z)}\left[\log \left(1-D_z(G(z))\right]\right.\end{aligned} \tag{2}$$

$$\begin{aligned}\mathcal{L}_{\text{GAN}}\left(F, D_y, x, z\right) = & \ \mathbb{E}_{z \sim p_{\text{ data }}(z)}\left[\log D_y(z)\right] \\ & + \mathbb{E}_{G(z) \sim p_{\text{ data }}(x)}\left[\log \left(1-D_y(F(G(z)))\right]\right.\end{aligned} \tag{3}$$

While equation 2 depicts the standard adversarial loss seen in a normal GAN, equation 3 represents the adversarial loss of the generated output, G(z), being used to generate fake input, F(G(z)), that can be be checked against actual input, z, by the new discriminator $\mathrm{D_y}$. Using the second pair of networks, the output from the original generator can be fed back into the new networks to try to reproduce the original input. Figure 4 illustrates how the process works. The fake output from the first pair of networks becomes the input of the second pair, using the original input as the new paired real data to produce a second set of real and fake data. By doing so, the system can determine a new concept referred to as cycle consistency [1].

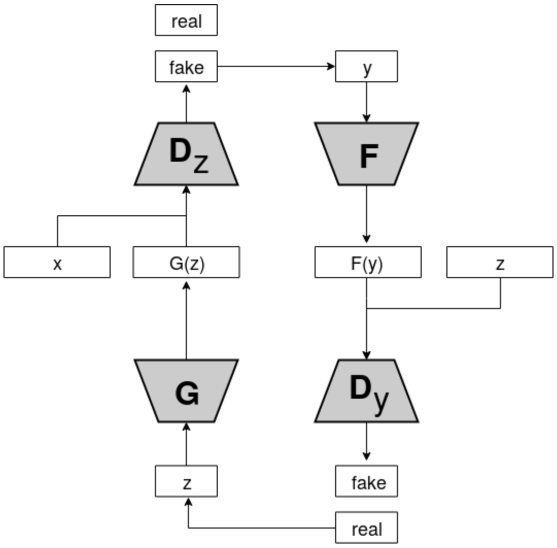

Figure 4: Cycle GAN architecture

Using cycle consistency, CycleGAN is able to also track the cycle consistency loss defined by the following loss function:

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) = &\, \mathbb{E}_{z\sim p\,\tan(z)}\left[\|F(G(z)) - z\|_1\right] \\ &+ \mathbb{E}_{y\sim p \text{ data } (y)}\left[\|G(F(y)) - y\|_1\right]\end{aligned} \tag{4}$$

Equation 4 tries to incentivize the behavior to maintain forward cycle consistency and backward cycle consistency by calculating the difference between the generated fake input, F(G(z)), and actual input, z, as well as generated fake output, G(F(x)), and real output, x [1]. By ensuring that the data is transitive, CycleGAN becomes an effective means to prevent mode collapse from occuring during training [1]. The total loss of the framework is then defined as the sum of each individual loss:

$$\begin{aligned}\mathcal{L}\left(G, F, D_X, D_Y\right) = &\, \mathcal{L}_{\text{GAN}}\left(G, D_Y, X, Y\right) \\ &+ \mathcal{L}_{\text{GAN}}\left(F, D_X, Y, X\right) \\ &+ \lambda\mathcal{L}_{\text{cyc}}(G, F)\end{aligned} \tag{5}$$

λ is used as a constant to control the importance of of the cyclic loss while the model is learning [1].

5.2 ProGAN

One suggestion that has been proposed for the stabilization of training GANs during image generation is through the progressive growing of layers, a technique labeled ProGAN [3]. The GAN begins by training on the images at a low resolution until convergence is reached, and then continues training at higher and higher resolutions in incremental steps. Not only does this aid in stabilizing the gradient of the generator while training, but also significantly improves the speed as the bulk of the converging step occurs at the lowest resolutions causing the time necessary to reconverge as additional features appear to be lowered [3].

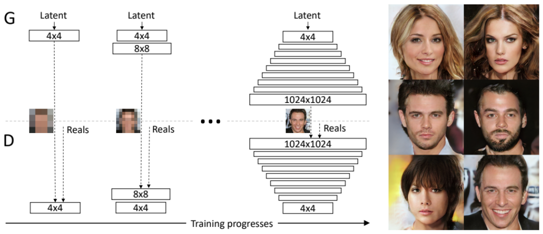

Figure 5: ProGAN method of learning on celebA dataset generation [3]

The process starts by downsampling the images passed as noise into the generator network as well as the real data that the output is compared with by the discriminator. As the model converges, the generator should be producing fake data directly comparable to real at that resolution. The generator then starts to upsample as an additional step to learn to produce the data at a slightly higher resolution while the discriminator again compares it to real data that has been downsampled slightly less. This continues until the generator has learned to produce data of the original resolution. Figure 5 depicts an example of this occurring using images of celebrities that have been downscaled to 4x4 pixel resolution which progressively is increased to the original size of 1024x1024 pixels during training.

5.3 WGAN

Wasserstein GAN (WGAN) makes use of the Wasserstein/Earth-Mover (EM) distance calculation when calculating loss, rather than the traditional adversarial loss described in section 4.6 to achieve the following loss function:

$$\min_{G} \max_{D\in\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_g}[D(\overline{\boldsymbol{x}}))] \tag{6}$$

In WGAN, the descriminator is referred to as the critic, as it is no longer used for classifying real or fake images, and instead uses the wasserstein loss to report to the generator the distance to the expected result [10]. Using this feedback, the generator can continue learning to improve regardless of whether the discriminator has already reached the optimal state or not. This has the desired effect of stabilizing the system allowing it to reach a convergence, removing mode collapse, as well as reducing the loss of a generator gradient [10]. Despite these benefits however, it still suffers from an issue where the weights of each node must be clipped to enforce the Lipschitz constraint. While this clipping is necessary to force convergence between the generator and discriminator to a single value; a large clipping parameter can significantly increase the time it takes for the critic to reach optimality while a small clipping parameter can cause the system to lose a larger portion of the generator gradient [10].

5.3.1 WGAN-GP

In order to solve this issue, WGAN-GP was proposed to eliminate weight clipping, and instead enforce the Lipschitz constraint through a gradient penalty that is added as an extra loss value:

$$L = \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})] + \lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}\left[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right] \tag{7}$$

Rather than clipping the weights to reach convergence, gradient penalty instead encourages the losses for the generator and discriminator to each approach a constant convergence through a two-sided penalty [11]. By allowing each network to still learn naturally while assisted by a penalty, the method avoids the pitfalls of the original clipping strategy.

5.4 ESRGAN

Enhanced Super Resolution GAN (ESRGAN) expands on the approach taken by SRGAN but making some additional improvements. They first expand their network structures to become much deeper, using what they introduce as a Residual-in-Residual Dense Block (RRDB) which they state has a higher capacity and improves its training ability [27]. They also remove any Batch Normalization in their networks in favor of residual scaling and smaller initializations to compensate for the deeper network. Additionally they make use of another published GAN implementation called Relativistic average GAN (RaGAN) for their discriminator which improves the information the generator recieves regarding texture details of the image [27].

To evaluate the effectiveness of ESRGAN, they used a no-reference quality metric used for single-image super resolution (Ma score) as proposed by Ma et al. combined with Naturalness Image Quality Evaluator (NIQE) such that:

$$\text{perceptual index} = \frac{1}{2}((10 - \text{Ma}) + \text{NIQE}) \tag{8}$$

Lower perceptual index shows better perceptual quality. This perceptual index was taken over the Root-Mean-Square Error (RMSE) for the final result. ESRGAN was additionally evaluated using Peak Signal-to-Noise Ratio (PSNR) in comparison to other known SR algorithms [27].

ESRGAN consistently showed better results with regards to PSNR compared to other known SR methods. ESRGAN also won first place at the 2018 PIRM-SR Challenge, competing for best super resolution algorithms, with the lowest perceptual index [27].

6 Comparison with other Projects

In this section, some previous implementations of GANs that performed the same tasks detailed by this paper will be examined. Each of these implementations saw some notable success at the task attempted and so will make a useful comparison for the potential improvements offered by the proposed methodology.

6.1 Cycle-Dehaze

Cycle-Dehaze is a particular implementation of CycleGAN modified for performing haze removal. An extra cyclic perceptual consistency loss was added to the model to allow a larger comparison of feature space rather than pixel space. The new loss function for Cycle-Dehaze is updated such that:

$$\begin{aligned}\mathcal{L}\left(G, F, D_x, D_y\right) = \mathcal{L}_{CycleGAN}\left(G, F, D_x, D_y\right) \\ + \gamma * \mathcal{L}_{Perceptual}(G, F)\end{aligned} \tag{9}$$

The original cycle consistency inherent in CycleGAN improves the PSNR while the new perceptual loss ensures the sharpness of the images used [28].

Cycle-Dehaze was tested on 1449 pairs of images using synthesized haze and increased that number further using data augmentation methods to produce modified versions of the original dataset. Their model was evaluated using PSNR and SSIM. These measures were compared against CycleGAN as well as the results of the NTIRE 2018 challenge on single image dehazing. Cycle-Dehaze was further tested on different datasets to check for overfitting on a singular dataset [28].

The results of this evaluation put Cycle-Dehaze above CycleGAN in PSNR and SSIM, but lower than the best results of the challenge. The author concedes that due to GPU limitations, their algorithm was forced to downscale images first before training and then re-upscaling them which can result in a loss of PSNR [28]. However, the results on cross-dataset testing was very promising, as the values of PSNR and SSIM were still high and even as good as CycleGAN on a single dataset. This shows the solution was not just overfitted to the challenge dataset, unlike many algorithms used for the challenge, and showed promise in being used on real world data.

7 Data

7.1 Collection

Data was collected for this task from the Tropospheric Emission Monitoring Internet Service website (TEMIS) [35]. This website, in cooperation with the European Space Agency (ESA), and hosted by the Dutch Meteorogical Institute (KNMI), provides a service for browsing and downloading atmospheric satellite data products. It gives access to products that consist of tropospheric trace gasses, aerosol concentrations, UV products, cloud information, and surface albedo climatologies [35]. These various sets of data come from different satellite instruments, including the OMI and TROPOMI measurement tools which are of interest for this paper. OMI and TROPOMI data were downloaded for the entire period between February 02, 2018 and July 06, 2019, providing roughly a year and a half of comparable OMI and TROPOMI data. This amount of data came out to roughly two terabytes of disk space. Each day contains roughly 14 and 12 OMI and TROPOMI files respectively, with each file pertaining to a swath of NO_2 measurement data recorded as left reflected off the surface and through the atmosphere as the satellite passed over the Earth in a southward arch between each poles.

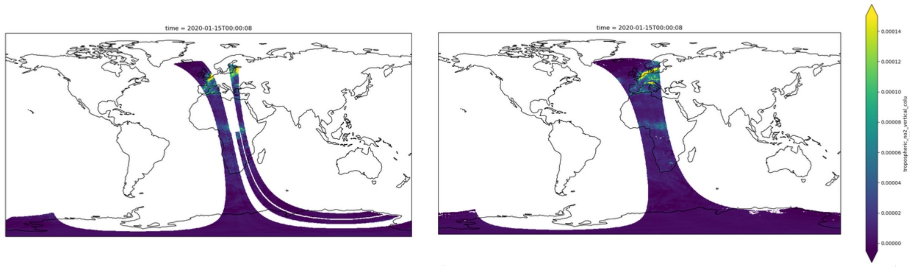
time = 2020-01-15T00:00:08
time = 2020-01-15T00:00:08
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
tropospheric_no2_vertical_colu
mn

Figure 6: Single swath taken from OMI satellite data (left) and from TROPOMI satellite data (right)

Altogether, these files compose a full coverage of the Earth's surface through the course of a single day. Measurment time occurs roughly over 100 minute increments, only taking measurements of the Earth in coordination with light from the Sun. Each file contains data in NetCDF format, which is a common method for storing atmospheric array oriented data. Within, are arrays representing the NO_2 values of the swath, corresponding latitude and longitudes, time at which each latitude is recorded by the measurement instrument, as well as various metadata such as quality, cloud coverage, and totals.

7.2 Challenges of the Data

While having this data was a useful starting point, there were many issues in getting started with processing it. For one, in order to fully leverage the learning capabilities of the GAN, a reliably paired dataset between OMI and TROPOMI was needed. However, in order to do so, each data pair must be examining both the same location and time to accurately compare features. To make this task even more difficult, the satellites recording OMI and TROPOMI data do not follow the same orbit, which means that regions recorded by both satellites aren't guaranteed to have been seen within a suitable timeframe of one another. Also, this limitation causes the raw data that does overlap and has been recorded at close to the same time to be skewed in representation compared to each other making it very difficult to select any specific area from both datasets. So of the two requirements for paired data, that of both temporal and geographical, neither are possible using the data in its raw form.

In addition, in order to train a machine learning model, a standard shape of input must be ensured across all the data that will be passed to the model. The data received from the satellite on the other hand, has no consistent shape, with the number of rows contained within each file varying widely. There also is not a precise difference in scale between the areas represented by OMI and TROPOMI, with some regions showing a difference in resolution along the x-axis by a factor of 7.5 all the way up to a factor of 8.5 in other regions. This variability is caused by the refraction of light through the atmosphere before reaching the sensor recording the data values. There are a multitude of different conditions in the atmosphere that can distort the light which not only affects resolution but also the latitude and longitude positions of specific data points. The latter effect also contributes to the difficulty in pairing the data as specific NO_2 features could shift by several kilometers between each satellite. The former effect of refraction also brings up another issue with training our model, specifically when attempting to perform a super resolution task, which is the awkward resolution scales that the model is attempting to improve between OMI and TROPOMI. The raw TROPOMI data showed an estimated resolution increase of two along the y-axis and eight in the x-axis. While it should be possible in theory to perform this kind of super resolution, it further complicates our task which is traditionally meant for improving the resolution of image data by a constant square scale.

One other point about the data, is that the OMI and TROPOMI data are not represented on the same unit scale. OMI is recorded in molecules per square centimeter while TROPOMI is recorded in mols per square meter.

8 Methodology

In this section, we discuss how each of the challenges in the data were adressed in order to create a working dataset for our model, as well as explaining the model being used for training and the steps taken in order to train the model.

8.1 Creating Trainable Data

In order to draw comparisons between the two datasets within our model, the unit scales must both be the same. An exact transformation is possible by applying a correction to our OMI data to change the values to be represented in mols per square meter just as TROPOMI is. This is handled by multiplying across our entire raw OMI dataset a correction value of 1.6605387831627262e-20.

As for the issues with comparing the data due to differences in orbit, shape, and scale; all these limitations should be solved at the same time by creating a pairable training set of data. To perform super resolution, we ideally want both the low resolution and high resolution data to mirror each other to a close extent. As mentioned earlier, since the satellites are following different orbits, points where the satellites overlap first needed to be found in order to create paired data. Time and location, both had to be taken into account before reliable pairs could be made. First, files were compared by the timestamp of the sensing period detailed in the file title, taking only files that were recorded within one hour of each other. This was done to ensure that the NO_2 values recorded for any region by both the OMI and TROPOMI satellites occurred close to the same time without much chance for features in the troposphere to move between satellites measurements.

In order to isolate overlapping areas, an atmospheric regridding tool called xESMF was used to translate the raw OMI and TROPOMI data to a standard shape and resolution scale. xESMF is a universal regridding tool for geospatial data written in Python that can perform geocentric regridding algorithms to transform data between curvilinear and rectilinear grids. In essence, our raw OMI and TROPOMI data is represented in differing curvilinear spaces which is the root of the issue in comparing them. Using xESMF, both were translated to a consistent rectilinear space separated by a constant scale factor of 4 using bilinear interpolation. This change gave us two matrices representing the world with the original NO_2 values accurately geolocated within a swath crossing some area inside of each grid. This generated two matrices of size 720x1440 and 2880x5760. The matrices were then split into smaller tiles of 32x32 and 128x128 that each contain the exact same area of the world for both the OMI and TROPOMI satellites respectively. Each pair was then checked to remove any tiles that did not contain 99% non-zero values which would represent areas outside of the original swath. This left only areas that fall within both swaths which were then saved as part of the training set. In addition, the areas around the poles were not saved as part of the paired data as the values were significantly stretched out compared to data closer to the equator due to coordinate-referencing issues.

While these steps created paired data matrices of the same location, close in time, and consistently shaped and scaled, it was still not enough to ensure that each pair contained similar features that could be recognized and used to perform super resolution. Also, even though the base files were recorded within an hour of each other, the actual sensing period for each file occurred over a longer period of time. This means that in actuality, if the same area was passed over by both satellites in the same sensing period, it could actually have occurred within a much larger difference in time, potentially causing one satellite to record entirely different features than the other. Although it was stated earlier that part of the reason NO_2 was chosen was for its slow feature changes, allowing these features to appear clearly in both OMI and TROPOMI recordings, since we are attempting to learn these features in small localized areas, the features can shift from one tile to another between each satellite recording. While part of the point of using a GAN for this task was to reduce the discrepencies caused by non-paired data using perceptual learning, it remains true that closer data pairs should allow the model to learn to reproduce features from OMI input more accurately. So in order to reduce this amount of non-paired data further, we made use of the variable in the original data detailing the time each latitude was recorded. First we obtained the latitude and longitude of the center point of each saved tile pair and located it's relative position in the original raw data file. Using this position, we could determine the exact time that latitude was crossed by each measuring satellite and once again remove any tiles that occurred with a difference of larger than an hour between them.

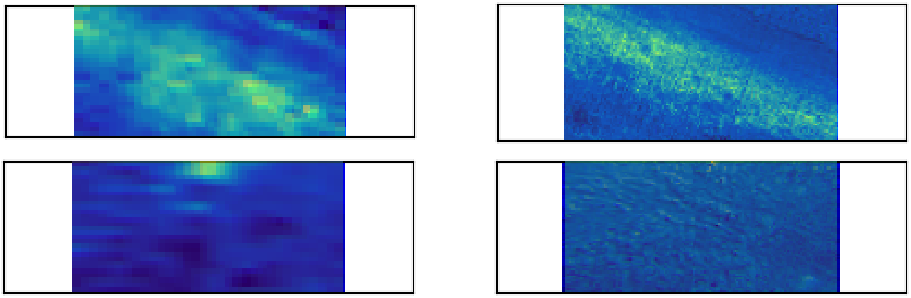

Figure 7: Paired data created from OMI (left) and TROPOMI (right). Top row depicts a visibly similar pair between OMI and TROPOMI while bottom depicts a pair without matching features.

All in all, through this method of processing the raw data files, roughly 20 thousand tile pairs were generated to be passed into the model for training.

8.2 Model

Out of the various different GAN types explored, we chose to use Wang, Xintao, et al.'s ESRGAN for this task as it is optimized for the super resolution of images, and so we expect similar results using our data. It was also selected due to its focus on perceptual loss while optimizing model weights. The ESRGAN was tested using regular satellite imagery which is closer to the type of data it was intended for, but also more unpaired comparable to the data we plan to use to train it with.

In order to prepare the model, several modifications were made to account for our datasets. First, the number of channels for the input and output to the networks were adjusted from three to one since it normally expects three channel RGB image data, with values occurring between 0 and 255, as opposed to our single channel NO_2 data, with values occurring between -2e^{-5} and 6e^{-5}. This should not negatively affect the training process as it is the same as performing super resolution as if we were using grayscale images of one channel. Second, we removed all the image processing steps as well as saving and loading of images. Because our data is quite precise, float values accurate to 6 decimal places, saving or processing the data as images in 255 pixel range would end up losing that extra precision. The default pretrained models for image data were removed so that the network could be retrained from scratch using these more precise values. The network was additionally set up to expect input of 32x32 pixel images with an upscale resolution of 4 to produce 128x128 pixel output images.

8.3 Training Process

To facilitate a smoother learning process from scratch, the model was trained in two phases. In the first phase, tile pairs were taken from the total set encompassing only those pairs which occurred within 10 minutes of each other as this set represents the most closely paired data between OMI and TROPOMI. There were approximately two thousand pairs in this set. A smaller subset of this data was taken out to be used as validation in tracking the training process. This validation set was composed of 100 pairs occurring within 10 minutes of each other from 4 different days over the course of the year and a half of data, one for each season. This was purposefully chosen as NO_2 readings will also vary by season but need to be capable of producing results regardless of the region or date. By starting with this training group, the model should first learn to recognize features existing in both the OMI and TROPOMI to learn what the ”correct” answer should be when generating upscaled OMI data. The model was trained over the course of 5 days utilizing 2 Nvidia Tesla K80 GPUs for 5000 epochs in total.

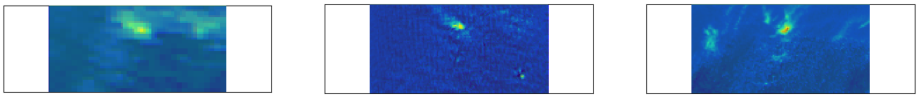

Figure 8: Set of images with OMI (left), generated TROPOMI (center), and original TROPOMI (right). Single example of a tile generated from an OMI and TROPOMI pair with alike features.

The second phase included the rest of the remaining data taken within one hour between each pair, bringing the total size of the training set to 20000 pairs. In addition to the previously used validation set, another 100 pairs were added, taken from one day during the summer but from pairs occurring within 20, 30, 40, 50, and 60 minutes of each other to view how the model handles different times of data. The model was trained for another week with this expanded dataset using the same GPUs for 1600 epochs.

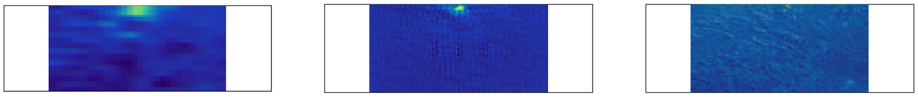

Figure 9: Set of images with OMI (left), generated TROPOMI (center), and original TROPOMI (right). Single example of a tile generated from an OMI and TROPOMI pair without matching features.

9 Results

9.1 Validation

OMI is traditionally validated by comparing it against TROPOMI values which were then validated against ground-truth values. For our validation, we take a similar approach by comparing the closeness of the entire swath of new generated TROPOMI data to the original TROPOMI with that of the original OMI. As the original TROPOMI data has already been validated, and the difference in time between features appearing in both swaths is minimal, we can test the validity of OMI and generated TROPOMI using original TROPOMI swaths as a model to the same extent that OMI is normally validated.

Two separate validation methods were tested while making this comparison. The first is the same method used traditionally by industry experts. This method involved taking the relative average of all overlapping pixels of OMI and TROPOMI falling within a localized area [36].

$$(OMI - TROPOMI)/TROPOMI \tag{10}$$

The mean value of this relative average over the area can then be taken to determine a rough similarity. A value closer to zero indicates more physical closeness between OMI and TROPOMI or generated TROPOMI and TROPOMI in the case of the new data. However, since OMI and TROPOMI paired swaths suffers from some feature translation due to weather, light refraction, and time, another more translation insenstive algorithm was tested as well. The Complex Wavelet Structural Similarity Index (CW-SSIM) is a variation of the standard image comparing SSIM method that separates images into multiple visual channels of waves which are more insensitive to consistent relative phase distortions [37]. In order to use this however, we first need to convert our data into images. This can easily be done by normalizing all the values in our data and then projecting it to a 255 based pixel scale.

To validate the model using these methods, 3 more dates of files were downloaded that weren't part of the original training set corresponding to days in summer, fall, and winter to test on. One file from each date containing a swath of data that passed over both Europe and Africa, due to clear presence of NO2 features in both hemispheres to test, were regridded to a rectilinear representation of the data. Each OMI matrix were then cut up into tiles and passed through the model before being pieced back together as a new generated TROPOMI matrix. The generated TROPOMI and the original TROPOMI were then downsampled to the same shape as the original OMI. Then, only the intersecting area between each pair of data, original OMI and original TROPOMI, and generated TROPOMI and original TROPOMI respectively, were isolated.

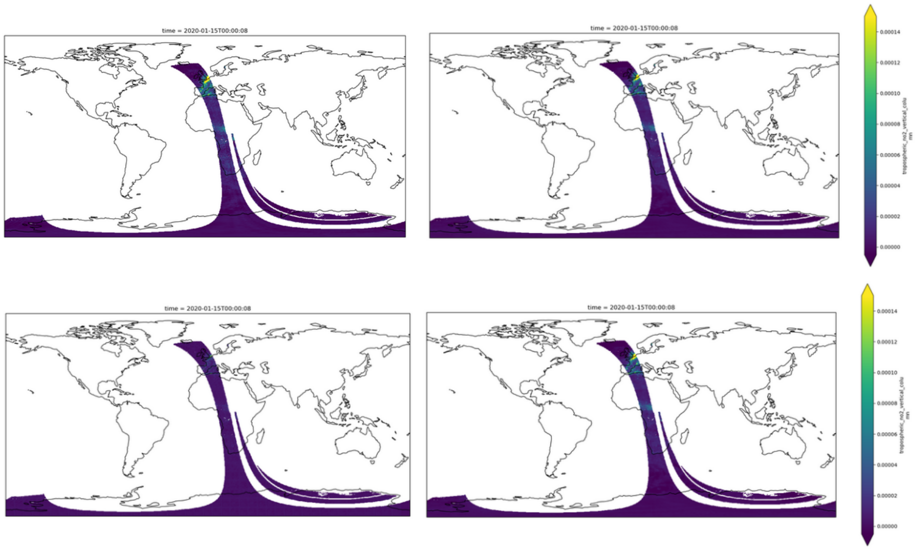
time = 2020-01-15T00:00:08
time = 2020-01-15T00:00:08
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
tropospheric_no2_vertical_colu mn
time = 2020-01-15T00:00:08
time = 2020-01-15T00:00:08
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
tropospheric_no2_vertical_colu mn

Figure 10: Intersecting area between validating OMI swath (left) and TROPOMI swath (right) occurring on 2020-01-15. Top row depicts original OMI and TROPOMI while bottom depicts generated TROPOMI and original TROPOMI.

Once these isolated paired regions of full swaths were obtained, the CW-SSIM was calculated between each.

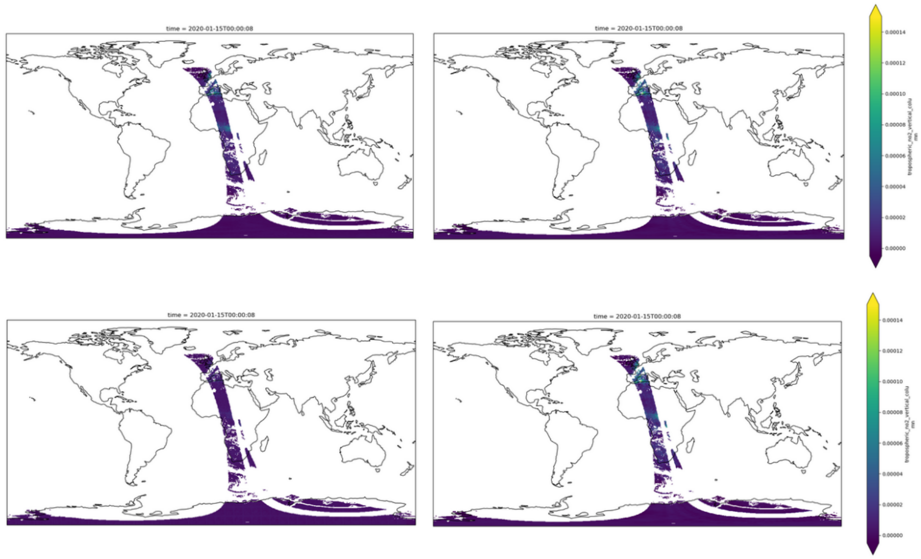
time = 2020-01-15T00:00:08
time = 2020-01-15T00:00:08
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
tropospheric_no2_vertical_colu mn
time = 2020-01-15T00:00:08
time = 2020-01-15T00:00:08
0.00014
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
tropospheric_no2_vertical_colu mn

Figure 11: Intersecting area between validating OMI swath (left) and TROPOMI swath (right) with the exclusion of TROPOMI data with accuracy of less than 75%. Top row depicts original OMI and TROPOMI while bottom depicts generated TROPOMI and original TROPOMI.

In addition to just comparing our results against TROPOMI, we also make use of a quality assurance variable in the TROPOMI data assessing the quality of each datapoint. We therefore test our results against both TROPOMI swaths as a whole, but also where TROPOMI is assured to have a determined quality of greater than 75

CW-SSIM	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	0.92	0.91	0.92	0.94
2019-10-15	0.93	0.93	0.97	0.90
2019-07-14	0.92	0.92	0.92	0.88

Table 1: Validation results for three separate swaths of NO_2 data following phase 2 of training occurring during summer, fall and winter periods taken after 1600 epochs of phase 2 training. Results are displayed over 4 columns representing a comparison of original and generated TROPOMI against validated TROPOMI values using CW-SSIM for both unaltered results and quality assured (QA) results.

Table 1 shows the values obtained after validating using CW-SSIM. As can be seen, our results have shown often on par or better than the original OMI values, with CW-SSIM values close to 1. However, while the accuracy taken over the swaths using the CW-SSIM method may appear to be high, this is infact misleading as it is mainly useful in feature comparison of images, which there is very little to be compared over the entire swath. The method used to traditionally verify OMI against TROPOMI is also performed on localized regions at a time rather than entire swaths. It is reasonable to perform our validation in the same way. Therefore, a smaller section of the swath was examined focused around Europe instead.

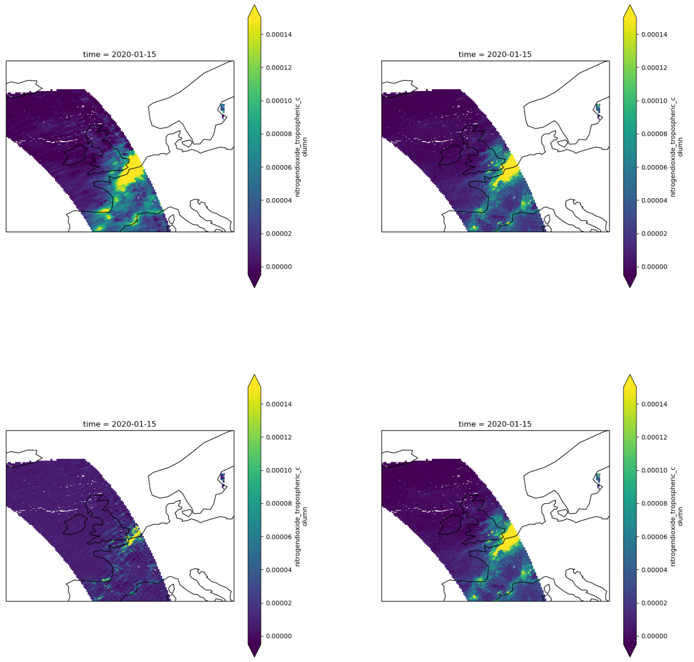

Figure 12: Intersecting area between validating OMI (left) and TROPOMI (right) over Europe on 2020-01-15. Top row depicts original OMI and TROPOMI while bottom depicts end-of-phase2 generated TROPOMI and original TROPOMI.

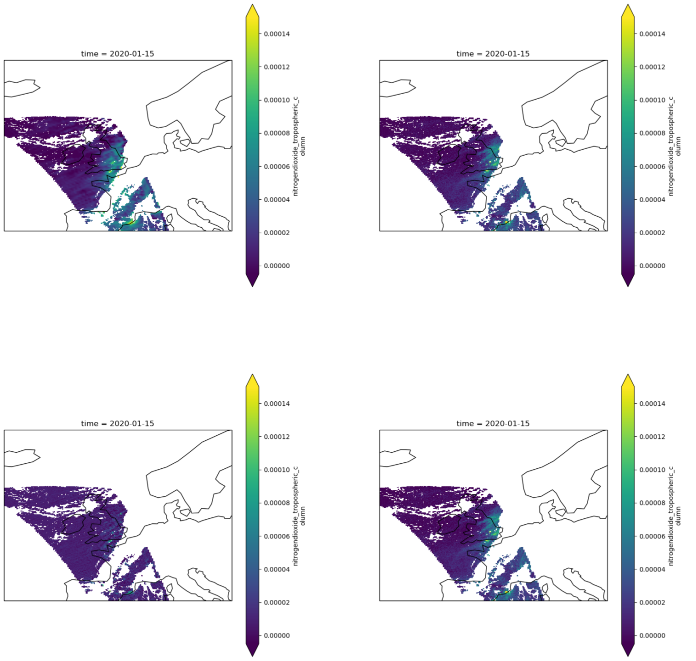

Figure 13: Intersecting area between validating OMI (left) and TROPOMI (right) subtracting QA values under 75% over Europe on 2020-01-15. Top row depicts original OMI and TROPOMI while bottom depicts end-of-phase2 generated TROPOMI and original TROPOMI.

We will validate our examples over Europe using both the CW-SSIM as discussed earlier to examine image similarity, as well as the physical validation method of the values of OMI and generated TROPOMI data against original TROPOMI used for traditional validation.

CW-SSIM	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	0.78	0.67	0.77	0.74
2019-10-15	0.78	0.50	0.86	0.53
2019-07-14	0.43	0.57	0.50	0.72
Rel. Avg.	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	-0.27	0.87	-0.45	2.47
2019-10-15	0.91	-0.69	0.72	-0.74
2019-07-14	-2.97	0.07	-1.34	-0.67

Table 2: Validation results for three separate dates in the European region of NO_2 data directly following phase one of training occurring during summer, fall and winter periods taken after 5000 epochs. Results are displayed over four columns representing a comparison of original and generated TROPOMI against validated TROPOMI values using CW-SSIM (top) and relative average (bottom) for both unaltered results and quality assured (QA) results.

CW-SSIM	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	0.78	0.65	0.77	0.70
2019-10-15	0.78	0.53	0.86	0.48
2019-07-14	0.43	0.48	0.50	0.63
Rel. Avg.	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	-0.27	-0.99	-0.45	-1.34
2019-10-15	0.91	-0.77	0.72	-0.77
2019-07-14	-2.97	-0.58	-1.34	-0.81

Table 3: Validation results for three separate dates in the European region of NO_2 data during phase 2 of training occurring during summer, fall and winter periods taken 400 epochs after phase 1 training. Results are displayed over 4 columns representing a comparison of original and generated TROPOMI against validated TROPOMI values using CW-SSIM (top) and relative average (bottom) for both unaltered results and quality assured (QA) results.

CW-SSIM	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	0.78	0.62	0.77	0.72
2019-10-15	0.78	0.45	0.86	0.47
2019-07-14	0.43	0.49	0.50	0.64
Rel. Avg.	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	-0.27	-1.12	-0.45	-1.87
2019-10-15	0.91	-0.53	0.72	0.40
2019-07-14	-2.97	-0.73	-1.34	-0.86

Table 4: Validation results for three separate dates in the European region of NO_2 data during phase 2 of training occurring during summer, fall and winter periods taken 900 epochs after phase 1 training. Results are displayed over 4 columns representing a comparison of original and generated TROPOMI against validated TROPOMI values using CW-SSIM (top) and relative average (bottom) for both unaltered results and quality assured (QA) results.

CW-SSIM	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	0.78	0.72	0.77	0.78
2019-10-15	0.78	0.57	0.86	0.69
2019-07-14	0.43	0.64	0.50	0.75
Rel. Avg.	Orig. OMI	Gen. Tro.	QA>75% OMI	QA>75% Gen.
2020-01-15	-0.27	3.31	-0.45	7.08
2019-10-15	0.91	-0.15	0.72	-0.01
2019-07-14	-2.97	0.22	-1.34	0.75

Table 5: Validation results for three separate dates in the European region of NO_2 data at the end of phase 2 training occurring during summer, fall and winter periods taken 1500 epochs after phase 1 training. Results are displayed over 4 columns representing a comparison of original and generated TROPOMI against validated TROPOMI values using CW-SSIM (top) and relative average (bottom) for both unaltered results and quality assured (QA) results.

Observing the results of these tables, it is apparent that generated TROPOMI likely never succeeds OMI in accuracy during the training period when compared against validated TROPOMI except in the test date of 2019-07-14 which will be discussed further in the Analysis section.

9.2 Analysis

While there is apparent accuracy in the results of generated TROPOMI that is clearly learning to draw features from given OMI input, it appears that this accuracy does not truly improve even after an extended period of training and may have even diminished with the inclusion of extra data pairs during phase two. This seems to be the case for two out of the three examples except for the case of 2019-07-14 which shows consistently higher accuracy of generated TROPOMI over OMI.

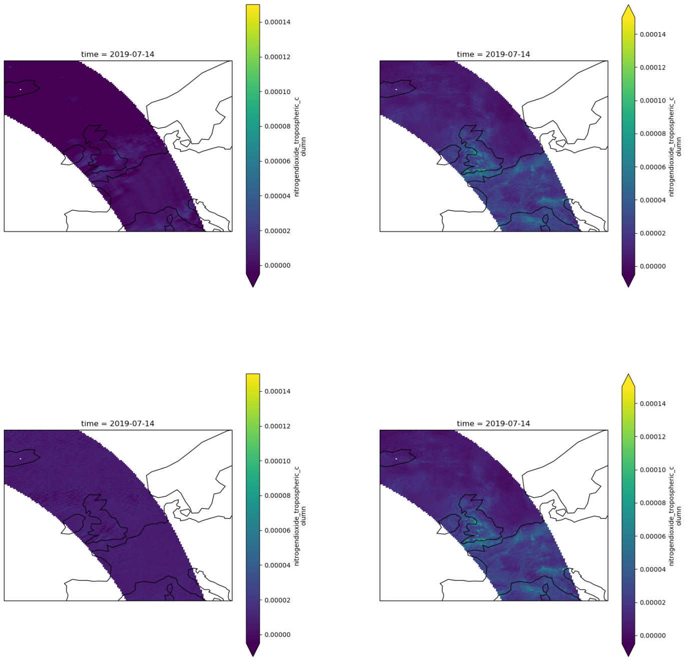

Figure 14: Intersecting area between validating OMI (left) and TROPOMI (right) over Europe on 2019-07-14 with a distinct lack of features occurring. Top row depicts original OMI and TROPOMI while bottom depicts end-of-phase2 generated TROPOMI and original TROPOMI.

In Figure 14, we see a ploted representation of the NO_2 data covering Europe on 2019-07-14. Differing from the other two examples, there is a clear lack of features present on this date. This suggests that the trained model might be better at producing featureless data from OMI input than that of features that we are actually interested in. The visualization of a generated TROPOMI example where features should be present, such as the display in Figure 12, also appears to be focusing more on background noise, and lessening the features present rather than accentuating them like we would prefer. These two observations coupled together, leads to a conclusion that the model is overfitted on featureless data. A quick search through the training set of TROPOMI examples for tiles containing values greater than 0.0002 (datapoints that might represent a peak within visible features) shows a count of 1700 tiles out of 20193 tiles total. The overwhelming number of potentially featureless tiles most likely occurring over areas of ocean, compared to actually featured data can easily account for how this overfitting could have occurred. This would also explain how the accuracy of the model seemed to slightly decrease during phase two when 18000 new tiles were added to the original 2000, increasing the division of featured and non featured tiles even more.

In order to determine what similarities of features could be drawn between original OMI, generated TROPOMI, and original TROPOMI, histograms were plotted of all three for examples where NO_2 features were clearly present.

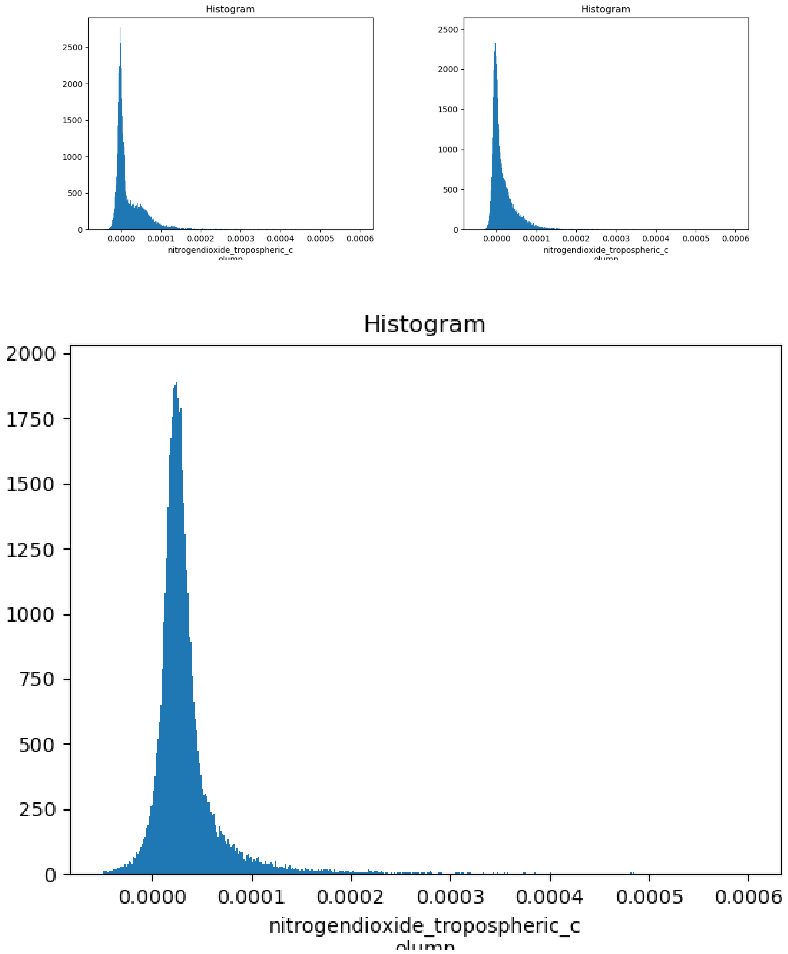

Figure 15: Histograms of NO_2 values of OMI (top left), TROPOMI (top right) and generated TROPOMI (bottom) taken on 2020-01-15 at end of phase 2 training.

Observing these drawn histograms in Figure 15, there appeared to be a min peak centered around zero, most likely representing background noise of a region, as well as max peaks likely occuring in the highest valued points of visible features past a threshold of 0.00013 where the histogram curves begin to flatten out.

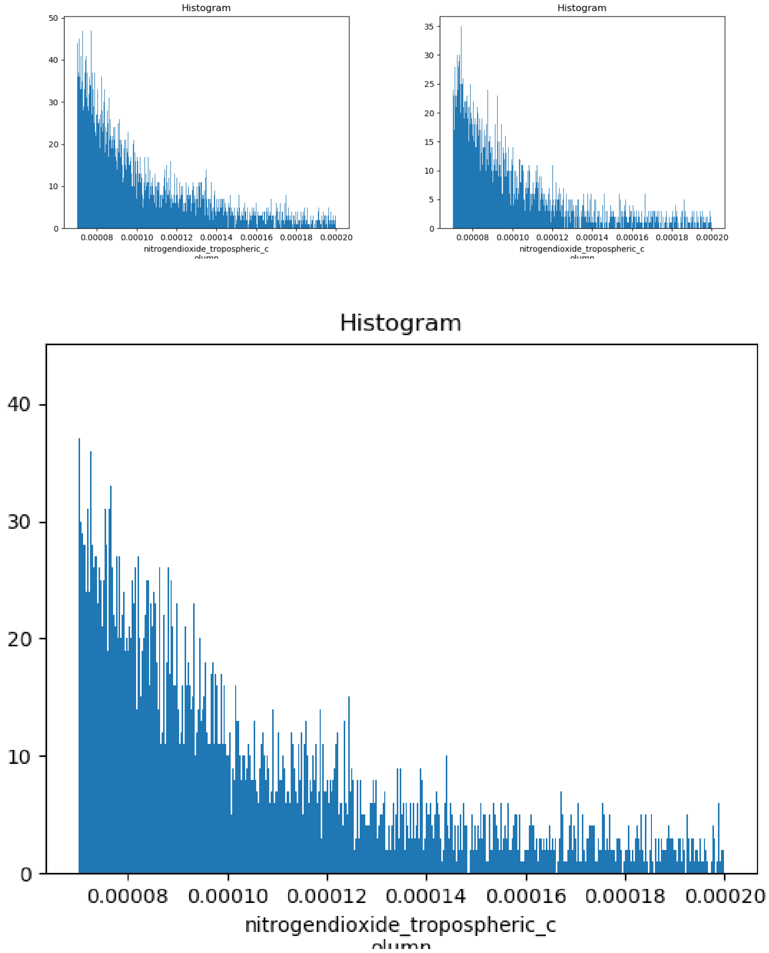

Figure 16: Histograms of NO_2 values of OMI (top left), TROPOMI (top right) and generated TROPOMI (bottom) taken on 2020-01-15 at end of phase 2 training isolating features falling within range 0.00007 and 0.0002.

Using this range of suspected features, illustrated in Figure 16, we can then try to isolate just the values representative of a feature within the region while removing external noise from each of the three. This was done by first subtracting 0.00007, the suspected min value of the feature range, from the data to set the features at zero relative, and then setting a threshhold at the new max of 0.00006 and min of 0.0 to fold in all values greater or lesser than those threshholds. The new values were then normalized over this range and then converted into an image as displayed in Figure 17.

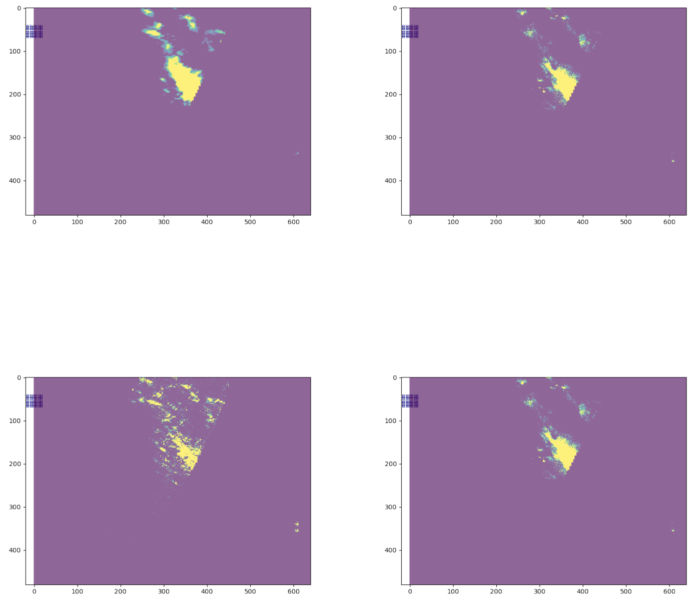

Figure 17: Intersecting area between OMI (left) and TROPOMI (right) over Europe on 2020-01-15 converted to image after isolating features. Top row depicts original OMI and TROPOMI while bottom depicts end-of-phase2 generated TROPOMI and original TROPOMI.

These newly created images should represent only the features recorded in OMI, generated TROPOMI, and TROPOMI while removing all background noise. As can be seen in these newly drawn images, the features in generated TROPOMI are actually very similar to the validated TROPOMI when highlighted without interfering background noise. A glance at the actual plotted values for OMI and TROPOMI back in Figure 12 shows that the shapes in these images are the same visible shapes present in those plots. Taking the CW-SSIM values once again on these normalized values gives us the results in Table 3.

CW-SSIM	5000 epochs	5400 epochs	5900 epochs	6500 epochs	Orig. OMI
2020-01-15	0.68	0.78	0.80	0.77	0.87
2019-10-15	0.60	0.65	0.66	0.62	0.78
2019-07-14	0.81	0.91	0.93	0.92	0.95

Table 6: Validation results for three separate swaths of NO_2 data across entire training period occurring during summer, fall and winter periods. Results are displayed over 4 columns representing a comparison of feature-only generated TROPOMI against validated TROPOMI values using CW-SSIM for both unaltered results and feature-only OMI and validated TROPOMI (last column).

This time, there is a clear and significant increase in the accuracy of generated TROPOMI between phase 1 and phase 2. While this might seem to contradict earlier conclusions drawn from the results, that is not necessarily the case. While it's true that many more non-feature tiles were added to the training set in phase 2, there were also many featured tiles that were also added. This means that the model could learn to draw more accurate features from these additional examples while still losing overall accuracy by reducing these features to a range of values closer to background noise. Isolating just these values shows that the features still persist in some form within the generated data.

As just an additional test, a constant factor of three was multiplied across the generated TROPOMI example to raise the value of all data points and display the potential features more clearly. We see in Figure 18 that this does indeed draw a picture closer to what we would expect.

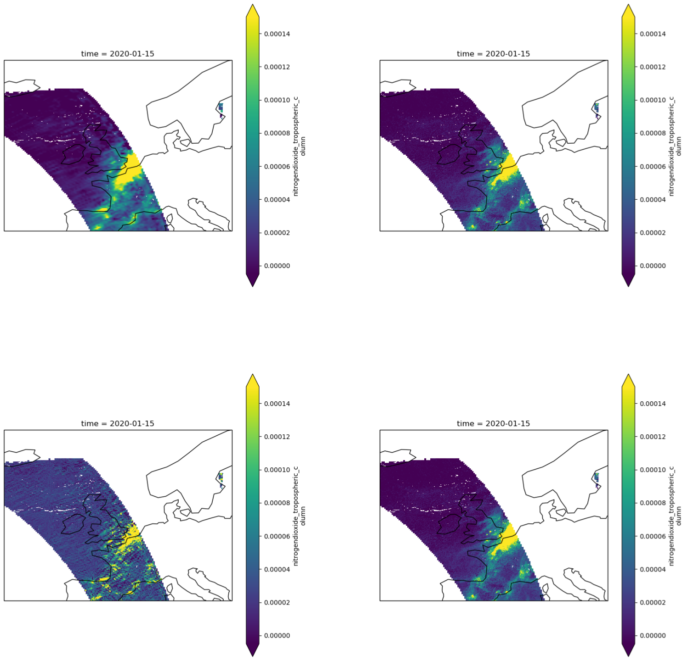

Figure 18: Intersecting area between OMI (left) and TROPOMI (right) over Europe on 2020-01-15. Top row depicts original OMI and TROPOMI while bottom depicts end-of-phase2 generated TROPOMI multiplied by a constant factor of three and original TROPOMI.

10 Conclusion

Through this paper, we have created a definitive way of creating trainable and paired data between volatile OMI and TROPOMI data. So long as both satellites pass over the same region with minimal enough change in time that NO_2 has not yet dispersed, we are able to create accurate geo-located tile pairs.

While it definitely looks possible to generate TROPOMI level data from OMI with a well defined training set, the model developed for this paper isn't yet successful enough to provide a replacement for OMI data. It appears that features clearly visible in both original TROPOMI and OMI data is much fainter and less defined in the generated TROPOMI. This is likely a result of the overwhelming majority of data pertaining to open ocean or sparsely populated landmass compared to dense urban areas that produce visible NO2 emissions. This sort of discriminate division in training data could definitely influence the model to favor creating less defined features as well.

If this is indeed the reason for the discrepancy between generated TROPOMI and original TROPOMI, it should be possible to retrain the model, choosing a more strategic division of featured and non featured-tiles, perhaps a 4:3 ratio or similar considering tiles containing features contain just as many data points without features. It wouldn't be prudent to only train using tiles containing features as an overfitting in the other direction would occur if the model learned that every tile requires features to be present to be valid generated TROPOMI. Therefore, some sort of division will still be necessary during subsequent training attempts.

Additionally, the CW-SSIM proved to be a useful method for measuring accuracy of OMI and generated TROPOMI against the validated TROPOMI data over local areas due to its recognizability of translations in image features. It is possible this method might prove to be more accurate than the traditional method used by industry experts to make relative average comparisons between OMI and TROPOMI.

10.1 Answering the Research Questions

Looking back on our originally defined research questions, we can now give some rudimentary answers for each.

> **1.** *Can GANs be leveraged to improve complex data relying mainly on perceptual learning?*
>
> We have seen that GANs can definitely improve the resolution of complex remote sensing data. Even with a large amount of training data that was completely unpaired, results still showed generated TROPOMI taking features directly from the OMI used as input. This shows that the GAN's perceptual loss is indeed feeding useful information to the model to help the learning process.
>
> **2.** *Can data collected from the low resolution OMI satellite be improved to the resolution of TROPOMI using GANs data enhancement techniques?*
>
> The data produced by the GAN certainly contained the sharpness and clarity of TROPOMI level resolution, however it still had not developed all of the features which should be present.
>
> **3.** *Will the accuracy of OMI data readings suitably improve when enhanced to TROPOMI level resolution?*
>
> While the accuracy of generated TROPOMI data did not overcome that of OMI on the whole, a displayed improvement in accuracy as a factor of useful training data was shown. This leaves open the possibility of overtaking OMI in accuracy with enough balanced examples to learn from.

10.2 Future Research

For future research, first and foremost another attempt should be made at training a model using a more calculated division of NO_2 heavy and sparse dataset.

It may also improve the training process to take the logarithmic value of the NO_2 data before passing it through the network. Normally, there is a very small difference between data that represents possible features, and areas absent of any features. By taking the log value of these areas, it should create a much larger gap between these patterns making potential features much more visible to the network and so also more reproducible. This could also help the model to differentiate NO_2 emissions from featureless areas reducing the need to strategically choose the samples to form a more varied dataset for training.

Another place that could be improved is the validation of generated data. There are only so much conclusions that can be drawn from validation that is twice removed from the actual ground-truth values. TROPOMI data itself isn't 100% accurate, nor is it equivalent to the OMI data that we are applying these methods to. It would provide a useful comparison in future research if generated TROPOMI data could also be validated against ground-truth values, which could then be measured against original TROPOMI in accuracy. While these methods were considered for this work, it would have proven to be a greater task than could be fit within the scope of this project. In addition, comparing the accuracy of CW-SSIM in estimating OMI similarity to TROPOMI with that of ground truth values could test the viability of CW-SSIM as a future testing metric of any new data collected.

As an aside, while this paper focused on leveraging the super resolution optimizations of the ESRGAN to perform this data enhancement, another suitable option to test would be the CycleGAN. CycleGANs have shown to be very capable in performing unsupervised learning with training data even more dissimilar to OMI and TROPOMI [1]. It might also be capable of solving the dissollution of features by the model as the second pair of networks must be capable of reproducing those features to maintain cycle consistency.

With the proof of concept provided in this thesis along with the visible potential for success, the ideas put forth here could also be applied in other areas of remote sensing with sensitive measurement instruments that are consistently being updated and improved, currently leaving obsolete older products behind.

References

[1] Jun-Yan Zhu, et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017.

[2] Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017.

[3] Karras, Tero, et al. "Progressive growing of gans for improved quality, stability, and variation." *arXiv preprint arXiv:1710.10196* (2017).

[4] Odena, Augustus, Christopher Olah, and Jonathon Shlens. "Conditional image synthesis with auxiliary classifier gans." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

[5] Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

[6] Goodfellow, Ian. "NIPS 2016 tutorial: Generative adversarial networks." *arXiv preprint arXiv:1701.00160* (2016).

[7] Guimaraes, Gabriel Lima, et al. "Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models." *arXiv preprint arXiv:1705.10843* (2017).

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

[9] Arjovsky, Martin, Soumith Chintala, and Lon Bottou. "Wasserstein gan." *arXiv preprint arXiv:1701.07875* (2017).

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

[11] Gulrajani, Ishaan, et al. "Improved training of wasserstein gans." *Advances in Neural Information Processing Systems*. 2017.

[12] Mroueh, Youssef, and Tom Sercu. "Fisher gan." *Advances in Neural Information Processing Systems*. 2017.

[13] Jain, Anil K., Jianchang Mao, and K. M. Mohiuddin. "Artificial neural networks: A tutorial." *Computer* 3 (1996): 31-44.

[14] Mino, Ajkel, and Gerasimos Spanakis. "LoGAN: Generating Logos with a Generative Adversarial Neural Network Conditioned on color." *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018.

[15] Barnett, Samuel A. "Convergence Problems with Generative Adversarial Networks (GANs)." *arXiv preprint arXiv:1806.11382* (2018).

[16] Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." *arXiv preprint arXiv:1812.04948* (2018).

[17] Sohn, Kihyuk, Honglak Lee, and Xinchen Yan. "Learning structured output representation using deep conditional generative models." *Advances in neural information processing systems*. 2015.

[18] Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017.

[19] Raissi, M., P. Perdikaris, and G. E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations." *Journal of Computational Physics* 378 (2019): 686-707.

[20] Hu, Zhiting, et al. "On unifying deep generative models." *arXiv preprint arXiv:1706.00550* (2017).

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

[22] Alpaydin, Ethem. *Introduction to machine learning.* MIT press, 2009.

[23] Zhou, Zhiming, et al. "Understanding the Effectiveness of Lipschitz-Continuity in Generative Adversarial Nets." (2018).

[24] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *nature* 521.7553 (2015): 436.

[25] Blau, Yochai, et al. "The 2018 PIRM challenge on perceptual image super-resolution." *Proceedings of the European Conference on Computer Vision (ECCV).* 2018.

[26] Ancuti, Cosmin, Codruta O. Ancuti, and Radu Timofte. "Ntire 2018 challenge on image dehazing: Methods and results." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.* 2018.

[27] Wang, Xintao, et al. "Esrgan: Enhanced super-resolution generative adversarial networks." *Proceedings of the European Conference on Computer Vision (ECCV).* 2018.

[28] Engin, Deniz, Anil Gen, and Hazim Kemal Ekenel. "Cycle-dehaze: Enhanced cyclegan for single image dehazing." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.* 2018.

[29] Freeman, William T., Thouis R. Jones, and Egon C. Pasztor. "Example-based super-resolution." *IEEE Computer graphics and Applications* 2 (2002): 56-65.

[30] Wang, Zhou, Eero P. Simoncelli, and Alan C. Bovik. "Multiscale structural similarity for image quality assessment." *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003.* Vol. 2. Ieee, 2003.

[31] Bre, Facundo, Juan M. Gimenez, and Vctor D. Fachinotti. "Prediction of wind pressure coefficients on building surfaces using artificial neural networks." *Energy and Buildings* 158 (2018): 1429-1441.

[32] Yu, Alex. How To Teach A Computer To See With Convolutional Neural Networks. Towards Data Science, Towards Data Science, 26 Nov. 2018, towardsdatascience.com/how-to-teach-a-computer-to-see-with-convolutional-neural-networks-96c120827cd1.

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

[34] Martin, Randall V. "Satellite remote sensing of surface air quality." Atmospheric environment 42.34 (2008): 7823-7843.

[35] European Space Agency. Tropospheric Emission Monitoring Internet Service. temis.nl

[36] Van Geffen, J. H. G. M., et al. "TROPOMI ATBD of the total and tropospheric NO2 data products." DLR document (2014).

[37] Sampat, Mehul P., et al. "Complex wavelet structural similarity: A new image similarity index." IEEE transactions on image processing 18.11 (2009): 2385-2401.