Hyejoon Lee

Generating Images of Aircraft Pressure Refuelling Scenes in Various Weather Conditions Using Image Style Transfer

School of Aerospace, Transport and Manufacturing
Computational Software of Techniques Engineering

MSc
Academic Year: 2023–2024

Supervisor: Dr Boyu Kuang
May 2024

School of Aerospace, Transport and Manufacturing
Computational Software of Techniques Engineering

MSc

Academic Year: 2023–2024

Hyejoon Lee

Generating Images of Aircraft Pressure Refuelling Scenes in Various Weather Conditions Using Image Style Transfer

Supervisor: Dr Boyu Kuang
May 2024

This thesis is submitted in partial fulfilment of the requirements for the degree of MSc.

Academic Integrity Declaration

I declare that:

- the thesis submitted has been written by me alone.
- the thesis submitted has not been previously submitted to this university or any other.
- that all content, including primary and/or secondary data, is true to the best of my knowledge.
- that all quotations and references have been duly acknowledged according to the requirements of academic research.

I understand that to knowingly submit work in violation of the above statement will be considered by examiners as academic misconduct.

Abstract

In this paper, we propose an unsupervised and unified multi-domain image-to-image translation model specifically for ground refueling image weather domain translation. By exploring existing methods that have demonstrated efficiency, we further address this challenging problem with our unique training and testing model. We investigate several critical design choices to enhance the effectiveness of contrastive learning in the image synthesis setting. Notably, our approach incorporates a diverse set of loss functions and feature extraction techniques. We demonstrate that our framework not only improves image quality but also reduces training time.

Keywords:
Generative adversarial networks, Weather image transfer, Ground refuelling system, Smart airport, Deep learning

Table of Contents

List of Figures

List of Tables

List of Abbreviations

SATM	School of Aerospace, Technology and Manufacturing
GAN	Generative Adversarial Networks

Chapter 1

Introduction

The aviation industry is undergoing a transformative shift towards implementing ”Smart Airports,” where the integration of highly automated systems is pivotal for achieving operational efficiency and ensuring safety standards. Among these systems, automated refueling stands out as a critical component, promising streamlined processes and reduced turnaround times for aircraft. The advent of smart airports aims to leverage cutting-edge technologies to enhance operational workflows, reduce human error, and optimize resource management. Automated ground refueling systems, in particular, play a vital role in this ecosystem by ensuring timely and precise refueling operations, which are essential for maintaining tight flight schedules and improving overall airport efficiency.

Figure 1.1: Ground Refuelling system

However, the successful deployment of automated refueling systems relies heavily on accurate and reliable sensing methods. Computer vision emerges as a cost-effective and safe sensing solution, enabling the ability to interpret visual data and autonomously make informed decisions. Computer vision systems are integral to identifying refueling ports, monitoring fuel levels, and ensuring secure connections between refueling equipment and aircraft. Yet, the performance of these systems is profoundly influenced by the prevailing visual conditions, which can undergo substantial alterations due to weather dynamics. These changes pose significant challenges to computer vision tasks, impacting crucial factors such as illumination, light sources, and background clutter.

Traditionally, addressing these challenges involved the laborious and time-intensive process of collecting a diverse range of visual data under various weather conditions. This

extensive data collection is necessary to train computer vision models that can generalize across different environmental scenarios. However, this approach is impractical and often insufficient for adequately training and testing computer vision models to operate effectively in real-world scenarios characterized by variable weather conditions. Collecting data for every possible weather condition is not only resource-intensive but also logistically challenging, especially for rare weather phenomena.

This project aims to bridge this gap by harnessing the capabilities of image generation techniques. By simulating a multitude of weather conditions, we can efficiently produce a comprehensive dataset of refueling scenes, essential for training and enhancing the accuracy of computer vision models in real-world, dynamic weather scenarios. Generative image techniques allow for the creation of realistic and varied visual data without the need for extensive real-world data collection. This simulated data can then be used to train models that are robust to a wide range of weather conditions, ensuring reliable performance in diverse operational settings.

Through the utilization of advanced methodologies such as image style transfer and Generative Adversarial Networks (GANs), this research endeavors to generate highly realistic images that faithfully replicate diverse weather conditions impacting aircraft pressure refueling scenes. By synthesizing such images, we can significantly mitigate the need for extensive field data collection efforts, thereby expediting the development and validation of robust computer vision models tailored for smart airport refueling systems. These generated images serve as a rich source of training data, enabling models to learn and adapt to the nuances of different weather conditions, ultimately leading to more reliable and efficient automated refueling systems.

Motivation and Overview

The motivation for this study is rooted in the need to enhance the operational efficiency and reliability of smart airport systems. Automated ground refueling is a critical process that can benefit significantly from advancements in computer vision technology. By improving the training data through generative image techniques, we aim to overcome the limitations posed by varying weather conditions and enhance the robustness of computer vision models used in automated refueling systems.

Aim

This study aims to develop a novel image generation model using ComboGAN to create realistic images of aircraft pressure refueling scenes under various weather conditions, thereby enhancing the accuracy and diversity of training data for weather-affected image recognition systems.

Current research mainly focuses on removing noise from a single weather image(2), such as removing fog(3) and rain, which is a one-way image translation. However, these approaches are only applicable to specific weather conditions and cannot be adjusted for multiple weather conditions flexibly. While training with diverse enhancements has been suggested by other papers, finding the optimal hyperparameters and model configurations remains a significant challenge.

Contributions

The contributions of this work are:

- To compare the result with the baseline model and demonstrate its improved efficiency in generating diverse and high-quality images.

- Finding the best approach to generate images in challenging weather conditions.

Chapter 2

Literature Review

In this section, we briefly introduce three closely related works, ground refuelling systems, image translation and generative adversarial networks.

2.1 Refuelling system

Aircraft refueling is a critical and highly regulated procedure involving several key steps to ensure safety and efficiency. The process typically starts with grounding both the aircraft and the fuel truck to prevent static electricity buildup, which can ignite fuel vapors. The fuel truck is parked strategically, usually at an angle, to facilitate quick evacuation in case of an emergency. Once positioned, a bonding cable is connected between the truck and the aircraft to equalize electrical potential and prevent sparks during fuel transfer.

Aircraft refueling occurs in varied environments, each with its own set of challenges. Refueling stations can be found at airports, which are designed to handle large volumes of fuel and frequent refueling operations. These stations must be equipped with safety systems like emergency shut-off valves and fire suppression equipment to manage potential accidents. Additional challenges include fluctuating weather conditions that can affect visibility, equipment functionality, and overall safety. For instance, heavy rain, fog, or snow can obscure vision and delay operations, while extreme temperatures can affect the performance of refueling equipment and fuel itself.

Based on this background, the integration of AI techniques into aircraft ground refueling systems is an emerging field aimed at addressing these challenges and enhancing efficiency, safety, and operational flexibility. AI can help to manage the above hazards by automating processes, improving decision-making, and increasing the precision of refueling operations. Several innovative technologies and AI applications are being developed and tested to automate and improve refueling processes for both ground and air operations.

Despite the advancements in AI and automation for aircraft refueling systems, several common limits and challenges persist across different implementations. Achieving the necessary precision and accuracy for autonomous refueling operations is technically challenging. AI systems must reliably detect and align with the refueling ports under various conditions, which requires advanced sensors and robust algorithms. Additionally, these systems must be resilient to environmental factors such as weather, which can in-

troduce variability and unpredictability into the refueling process. For example, sensors and vision systems must be able to function accurately in rain, fog, or snow, which can obscure visual cues and affect sensor readings.

Recent advancements in computer vision techniques have highlighted the importance of incorporating environmental factors, such as weather conditions, into automated fueling systems. Image transfer of weather conditions emerges as a crucial component in enhancing the efficiency and precision of these processes. By simulating various weather scenarios through image transfer techniques, such as style transfer and Generative Adversarial Networks (GANs), automated fueling systems can adapt to changing environmental conditions with greater accuracy and reliability.

The generative model can be used to generate more training data in situations where the training set is insufficient. Many of the structures used in computer vision require increasingly large datasets, especially those based on vision transformers, which have become increasingly popular in recent years and require a much larger amount of training than traditional CNN-based models. However, human classification is time-consuming and labour-intensive, and there are practical limitations. By generating synthetic data that mimics real-world conditions, these models can be trained more effectively, leading to improved performance and robustness in diverse operational environments.(4)

2.2 Genertative model

A generative model is a model that learns a given training data and produces similar data that follows the distribution of the training data, i.e., it produces new samples with the same distribution as the given training data. Most of the problems in computer vision that we have discussed so far are based on general classification, which is the problem of finding a classifier in feature space that can distinguish between the labels we are looking for.

However, the goal of a generative model is not to find a classifier, but to learn the distribution of a training set. Instead of labelling each piece of data, it learns the joint distribution of all the data. The goal of a generative model is to produce data that doesn't actually exist in the world, but feels like it does, especially between the data we've trained it on. It trains so that real image A and generated image B are similar. Here, B is the distribution of generated samples generated by the generative model, and A is the distribution of training data, i.e., data from the real world. There are two broad categories here Explicit Density Estimation (ex. VAE, Approximate Density, etc.), which clearly defines the generated image B, and Implicit Density (ex. GAN, Markov Chain, etc.), which generates samples from the generated image B without defining the model.(5)

The problems solved using generative models can be summarised into three categories.

For example, let's say we're building a generative model that produces weather images based on a weather cue. We want to make sure that the distribution of the model $p_\theta(X)$ is closest to the real data $p(X)$. So how do we do this? We can define the problem as minimising the distance between the model distribution $p_\theta(X)$ and $p(X)$.

1. Density estimation

   Can we find the probability $p_\theta(X)$ assigned by the model for a given datapoint x?

2. Sampling

   Can we create an image with a model distribution that does not exist in the training, but represents the distribution most similar to the training?

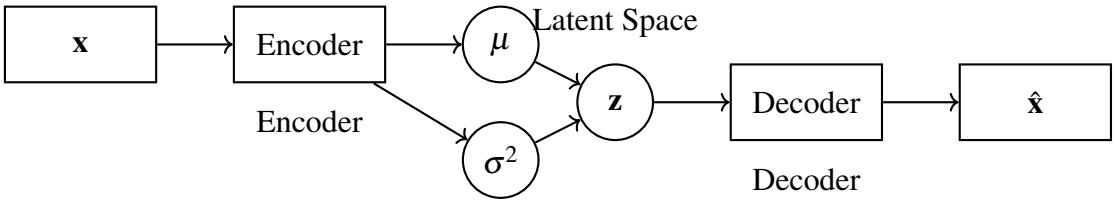
3. Unsupervised representation learning

   Can we learn a meaningful feature representation from a specific datapoint x?

The nature of the problems that generative models solve makes quantitative evaluation non-trivial. While there are some quantitative metrics, especially for sampling or representation learning, they do not reflect the qualitative nature of the images produced or the features found. Developing quantitative evaluation methods for generative models is therefore a very active area of research, and not all generative models are efficient and perform well in all situations. Even GANs, which are currently in vogue, have a very limited set of situations in which they can be used.

Given the diverse applications and the varying strengths of different generative models, the following sections focuses on two representative technologies: Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs). VAEs are selected for their capability in explicit density estimation and their robustness in learning latent representations. They are particularly useful for tasks that require a clear probabilistic interpretation of the data distribution. On the other hand, GANs are chosen for their powerful implicit density capabilities and their strength in generating highly realistic samples. GANs have revolutionized the field of image synthesis with their ability to produce images that are often indistinguishable from real photographs. By examining these two technologies, we can cover a broad spectrum of generative modeling approaches and highlight their respective advantages and limitations.

2.2.1 VAE



The Variational Autoencoder (VAE) is a powerful generative model that seeks to maximize the likelihood of observing the desired outcome x given the model parameters θ. The probability $p_\theta(x)$ indicates how well the model captures the underlying data distribution, with higher values suggesting a better model. In the context of VAEs, the goal is to learn the parameters θ such that $p_\theta(x)$ is maximized.(6)

One of the key advantages of VAEs is their ability to handle intractable densities, which would otherwise be difficult to compute. By employing variational inference, the VAE approximates these densities and derives a lower bound, known as the Evidence Lower Bound (ELBO), which is optimized during training. The VAE architecture consists of two main components: the encoder and the decoder. The primary objective lies in the decoder, which reconstructs the input data from a latent representation. The encoder is

introduced to facilitate the training of the decoder by mapping the input data into a latent space.

Unlike traditional Autoencoders (AEs), which aim to reconstruct the input data from a deterministic latent code, VAEs extend this approach by representing the latent code as a probabilistic value based on a Gaussian distribution. The VAE model learns to generate a latent space that is more densely packed towards the center, leading to better performance in reconstructing the original data compared to standard AEs.

Mathematically, the intractable density $p_\theta(x)$ can be expressed as:

$$p_\theta(x) = \int p_\theta(x \mid z) p_\theta(z)\, dz$$

where z is the latent variable. The ELBO, which the VAE maximizes, is given by:

$$\mathscr{L}(\theta, \phi; x) = E_{q_\phi(z|x)} \left[\log p_\theta(x \mid z)\right] - \mathrm{KL}\left(q_\phi(z \mid x) \parallel p_\theta(z)\right)$$

This lower bound ensures that the model captures the essential structure of the data while making the latent space more regularized. Consequently, the VAE outperforms traditional AEs, especially in generating new data points and reconstructing the original data from the latent space.

2.2.2 GAN

To understand the basic philosophy of GANs, the problem is: 'I want to sample from a complex, high-dimensional training distribution.' To solve this problem, the strategy is to sample a simple distribution (e.g. random noise) and learn parameters that can be transformed to follow the training distribution. The Generator's job is to learn the distribution of the real data and create fake data. $\rightarrow$ The goal is to make it as confusing as possible for the Discriminator.(7)

The Discriminator is responsible for distinguishing whether the sample is real data (training) or not. $\rightarrow$ The goal is to make the fake image as confusing as possible.

Discriminator's Objective Function

The discriminator in a GAN can be thought of as a classifier that returns 1 if the input data x is real and 0 if the input data is fake. Let $D(x)$ represent the output of the discriminator. The objective function of the discriminator can be defined as follows:

$$\max_D E_{x \sim p_{\text{data}}(x)} \left[\log D(x)\right] + E_{z \sim p_z(z)} \left[\log(1 - D(G(z)))\right]$$

In this equation:

- $D(x)$ is the discriminator's estimate of the probability that x is a real data point.
- $G(z)$ is the generator's output when given a noise vector z as input.

When the input is a real image, if the discriminator correctly predicts it as real, $D(x) = 1$, leading to $\log(D(x)) = 0$. However, if the discriminator incorrectly classifies a real image as fake, $D(x) = 0$, resulting in $\log(D(x)) = -\infty$. Conversely, if the input is a fake

image and the discriminator wrongly classifies it as real, $D(x) = 1$, so $\log(1 - D(x)) = -\infty$. If the discriminator correctly identifies the fake image, $\log(1 - D(x)) = 0$.

The objective function for the discriminator ranges from a maximum of 0 to a minimum of $-\infty$. The goal is to have a discriminator that always predicts accurately, achieving the maximum value of 0. To achieve this, the discriminator must be trained using gradient ascent, aiming to maximize its performance.

Generator's Objective Function

Now, let's consider the generator's objective function. Unlike the discriminator, the generator's focus is not on how well the discriminator performs on real images, but rather on how much it can confuse the discriminator with fake images. Thus, the generator's objective function is derived from the discriminator's response to fake images:

$$\min_G E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

Here, the generator aims to minimize this function by generating images that the discriminator classifies as real. This objective function, like the discriminator's, has a range from $-\infty$ to 0. However, in this case, the generator is trained using gradient descent, as the goal is to minimize the loss by generating more realistic fake images.

GAN's Overall Objective Function

The overall objective function of the GAN is the combination of the discriminator's and the generator's objective functions. The discriminator seeks to maximize its function, while the generator seeks to minimize its own. The GAN objective function can be written as:

$$\min_G \max_D E_{x \sim p_{\text{data}}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

This formulation illustrates the adversarial nature of GANs, where the generator and discriminator are in a continuous game, with the generator trying to produce more convincing fake images and the discriminator trying to distinguish between real and fake images.

2.2.2.1 Adversial Learning

$$\mathscr{L}_{\text{GAN}}(G, D, X, Y) = E_{y \sim Y}[\log D(y)] + E_{x \sim X}[\log(1 - D(G(x)))]. \tag{2.1}$$

(8)

2.3 Different GANs

Image transfer techniques, including Generative Adversarial Networks (GANs) like StarGAN and CycleGAN, offer avenues for simulating diverse weather conditions.

StarGAN,(9) a unified GAN framework for multi-domain image-to-image translation, demonstrates scalability and flexibility by training across multiple datasets with different

domains simultaneously. It utilizes a single generator and discriminator to handle multiple domain translations, incorporating domain labels into the training process for enhanced performance. StarGAN's efficacy in tasks like facial attribute transfer underscores its potential for facilitating realistic weather simulations in fueling systems.

CycleGAN,(10) another GAN-based approach, focuses on learning mappings between two image domains without requiring paired training data. While originally designed for tasks like style transfer and image-to-image translation, CycleGAN's ability to learn domain mappings in an unsupervised manner could prove beneficial for generating diverse weather conditions in automated fueling processes. However, challenges such as ensuring realism and mitigating artifacts require careful consideration in their application.

ComboGAN

2.3.0.1 Decoupling the Generators

ComboGAN introduces an innovative approach to address the computational costs associated with multi-domain image translation models. In traditional models, adding a new domain typically requires the addition of multiple new networks, which can significantly increase the computational load. To mitigate this, ComboGAN decouples the generator networks and domains from one another, streamlining the architecture.(11)

The generator networks in ComboGAN are derived from those used in CycleGAN but are restructured to enhance efficiency. Specifically, each generator is split into two components: an encoder and a decoder. The encoder is responsible for extracting domain-specific features from the input image, while the decoder reconstructs the image in the target domain. This modular design allows for a flexible combination of encoders and decoders to facilitate domain translation.

For instance, when translating an image x from domain A to an image y in domain B, the process can be represented as y = G_BA(x) = Decoder_B(Encoder_A(x)). This structure not only reduces the number of required networks but also enables computational efficiency. The output of Encoder_A(x) can be cached and reused for translating the image into other domains, thereby avoiding redundant computations.

With this approach, ComboGAN requires only one generator (an encoder-decoder pair) per domain, resulting in a linear scaling of the number of generators with the number of domains. This is in contrast to traditional models, where the number of generators increases quadratically with the addition of new domains. The discriminators in ComboGAN, however, remain domain-specific and scale linearly, as each domain retains its own discriminator. This architecture significantly reduces the computational complexity while maintaining the model's flexibility and effectiveness in multi-domain image translation tasks.

- ResNetEncoder

Component	Description
Initial Layer	
ReflectionPad2d	Pads the input tensor using reflection of the input boundary.
Conv2d	Applies a 2D convolution over the input.
InstanceNorm2d	Applies instance normalization.

PReLU	Parametric ReLU activation function.
Downsampling Layers	
Conv2d (Layer 1)	Reduces spatial dimensions by half and increases the number of feature maps.
InstanceNorm2d (Layer 1)	Normalizes the output.
PReLU (Layer 1)	Activation function.
Conv2d (Layer 2)	Further reduces spatial dimensions and increases the number of feature maps.
InstanceNorm2d (Layer 2)	Normalizes the output.
PReLU (Layer 2)	Activation function.
ResNet Blocks	
ResNet Block 1	Series of convolutional layers, normalization, and PReLU activations.

- ResNetDecoder

Component	**Description**
ResNet Blocks	
ResNet Block 1	Series of convolutional layers, normalization, and PReLU activations.
Upsampling Layers	
ConvTranspose2d (Layer 1)	Increases spatial dimensions by a factor of two and reduces the number of feature maps.
InstanceNorm2d (Layer 1)	Normalizes the output.
PReLU (Layer 1)	Activation function.
ConvTranspose2d (Layer 2)	Further increases spatial dimensions and reduces the number of feature maps.
InstanceNorm2d (Layer 2)	Normalizes the output.
PReLU (Layer 2)	Activation function.
Final Layer	
ReflectionPad2d	Pads the input tensor using reflection of the input boundary.
Conv2d	Applies a 2D convolution over the input to produce the final output.
Tanh	Applies the Tanh activation function to scale the output to the range [-1, 1].

They invented the plexer concept enables the creation of separate networks for each domain while sharing some parameters, which can be beneficial for tasks requiring domain adaptation. The discriminator is based on the PatchGAN architecture(12), making it suitable for image to image translation tasks. Plexter is an abstract class that handles common functionalities for both the generator and discriminator pleaters. Especsially,

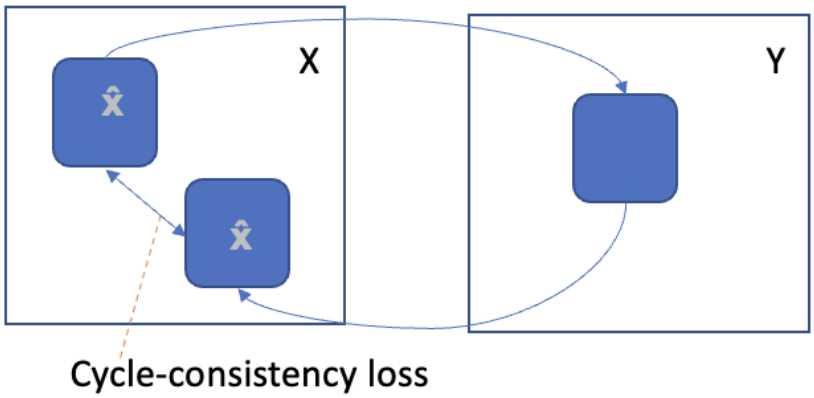


Figure 2.1: Cycle Consistency Loss

G_Plexer extends this class to manage the multiple encoder-decoder pairs for the generator. It also handles shared blocks if they exist and manages the forward pass through the encoder and decoder networks.

2.3.0.2 Shared Layers

ResnetGenShared is a sophisticated module designed for shared feature extraction and processing across multiple domains. It incorporates dynamic context handling to adjust processing based on domain-specific needs, making it flexible and efficient for multi-domain tasks. Each residual block processes the input features and learns to identify patterns and structures that are common across different domains. This is achieved through the convolutional layers that capture spatial hierarchies and textures.

2.3.0.3 Consistency Loss

CycleGAN introduces the concept of cycle consistency for training with unpaired data. This concept suggests that if the generators G and F are effective, mapping a real image x from domain X to domain Y and then back to X should yield the original image x. Mathematically, this is expressed as $x \to G(x) \to F(G(x)) \approx x$. Similarly, for an image y in domain Y, the reverse mapping should return y, i.e., $y \to F(y) \to G(F(y)) \approx y$.

Figure2.1 graphically shows the idea of cycle consistency, which is enforced through the following cycle consistency loss.

$$\mathscr{L}_{\text{cyc}}(G,F) = E_{x\sim X}[\|F(G(x)) - x\|_1] + E_{y\sim Y}[\|G(F(y)) - y\|_1]. \tag{2.2}$$

2.3.0.4 Problems

Cycle consistency encourages generators to avoid unnecessary changes and thus to generate image that share structural similarity with inputs. No matter how generate image look

like regenerated image should look like original image by reconstruction the generated image to original domain. Cycle consistency also prevents generators from excessive hallucinations and mode collapse, both of which will cause unnecessary loss of information. But It has unrealistic artefacts since its too restrictive occasionally. Cycle consistency is enforced at pixel level which led to unrealistic artefacts.

cycle-consistent image translation frameworks have been observed to obscure reconstruction details within subtle noise, making them difficult to detect. Additionally, the underlying assumption of cycle consistency—that the mapping between two domains is a bijection—can be overly restrictive and may not accurately reflect the complexity of real-world relationships between these domains. (13)

To further stabilize the training process and improve the quality of the generated images, several solutions have been adopted that seem to fit the project.

2.3.0.5 Different Approaches

When discussing the challenges associated with cycle consistency loss, numerous research efforts have been dedicated to addressing these issues. These studies typically fall into two broad categories: enhancements to the CycleGAN framework (14) (15) (16)(17). and approaches that forgo the use of cycle consistency loss altogether (18) (19) (20).

The first category includes various strategies to improve the CycleGAN architecture by integrating additional loss functions, introducing robustness measures, or proposing new training techniques. Conversely, the second category explores alternative models that achieve unpaired image-to-image translation without relying on cycle consistency, employing methods such as distance-based learning, geometry consistency, or contrastive learning techniques.

2.3.1 Self attention in GAN

Self-attention has been widely adopted in GAN models to address the challenges associated with generating high-quality images, particularly in capturing long-range dependencies and intricate details. Traditional GANs often struggle with these issues because they rely heavily on convolutional layers that are inherently limited to local spatial features. Self-attention, however, provides a powerful mechanism to overcome these limitations by allowing the model to selectively focus on different regions of an image, regardless of their spatial distance. This is achieved through attention mechanisms that calculate the relevance of each part of the image in relation to others, enabling the network to build a more coherent global representation.(21)

The introduction of self-attention in GANs, as exemplified by the Self-Attention GAN (SAGAN)(21), allows the model to effectively combine local and global information. This enhancement leads to improved image generation, where both fine details and overall structure are better preserved. Overall, the main motivation for adopting self-attention in GANs is to enhance the quality and diversity of the generated images, especially in tasks requiring high-resolution image synthesis.

Chapter 3

Methodology

3.1 DataSet Configuration

To train and evaluate our model, we utilize the BDD100K dataset(22), which has been extensively validated in prior GAN research, alongside a curated subset of the AGR dataset, specifically focusing on scenes depicting aircraft ground refuelling systems or similar scenarios. This combination of datasets ensures a comprehensive representation of weather conditions that impact aircraft ground refuelling operations.

3.1.1 Dataset Sources

BDD100K: A large-scale, diverse driving video dataset widely recognized and validated in GAN-related research. It provides a robust foundation for training generative models, particularly in the context of varying environmental conditions.
AGR Dataset: This dataset contains specific scenes of ground refuelling systems, which are critical for our application. The inclusion of these images is essential to accurately simulate the operational environment of aircraft refuelling under different weather conditions.

3.1.2 Image Preprocessing

All images are resized to 256×256 pixels to strike a balance between computational efficiency and image quality during the training process. This resizing ensures that the model can process the images effectively while maintaining sufficient detail for accurate weather condition simulation.

3.1.3 Weather Conditions

The dataset encompasses images representing five distinct weather conditions, each critical for robust model training across various scenarios:

- Clear
- Cloudy

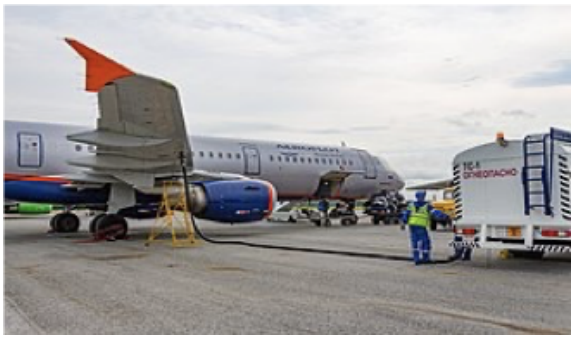


Figure 3.1: Example Image from AGR Dataset

- Rainy
- Snowy
- Foggy

3.1.4 Data Composition

Each weather condition domain consists of approximately 300-500 images, providing a substantial dataset for both training and testing purposes.

3.1.5 Data Splitting

To facilitate effective model training and unbiased performance evaluation, the dataset is split into:

- **Training Set:** 80% of the images, used to train the model and optimize its parameters.
- **Testing Set:** 20% of the images, reserved for evaluating the model's generalization and performance on unseen data.

By leveraging a combination of the BDD100K and AGR datasets, resized to a manageable input size and encompassing multiple weather conditions, our model is well-equipped to handle the complexity and variability inherent in generating realistic weather-transferred images for the aircraft ground refuelling system.

3.2 Model Architecture

Fig3.2 illustrates the architecture of our proposed method, which builds upon the ComboGAN framework. This architecture is further detailed in the subsequent sections.

In this work, we address the challenge of weather translation, particularly in scenarios where paired images of the same scene under different weather conditions are scarce. We approach this task as an unsupervised image translation problem. When transforming the entire image, our method focuses on enhancing structural consistency while preserving other elements of the scene. The network is designed to tackle three key tasks essential to

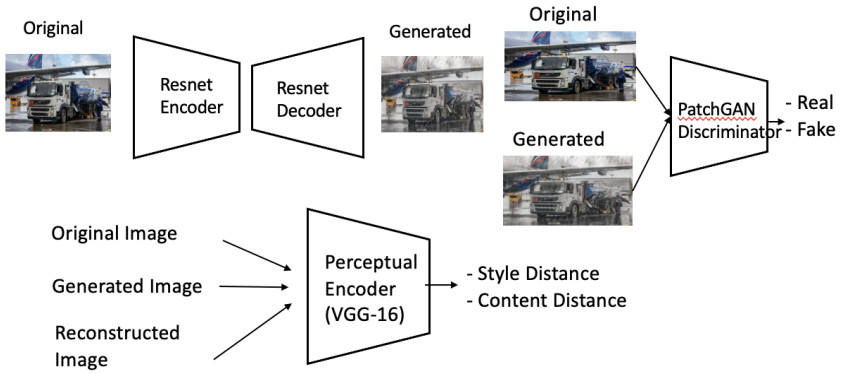


Figure 3.2: The overall architecture of the proposed method based on ComboGAN (a) The proposed shared generator architecture, (b) a PatchGAN Discriminator, (c) Perceptual Encoder(1) (bottom)

successful weather translation: (1) accurately identifying objects within the image, (2) effectively capturing and interpreting weather-specific cues, and (3) ensuring that unrelated areas of the image remain unaffected during the translation process.

3.2.1 Architecture Details

3.2.1.1 Generator Encoder-Decoder Structure

The generator consists of a shared encoder that learns to map images from any domain into a shared latent space. Each domain has a separate decoder that maps the latent representation to the target domain. During training, the encoder learns to capture the essential features of images from various domains, while the decoders learn to reconstruct images in their respective domains from the shared latent space.

Latent Space Mapping: Images from different domains are encoded into a common latent space. The latent representations are then decoded into images of the target domains using the corresponding decoders.

3.2.1.2 Discriminator

Instead of the global image discriminator, it is employed with the PatchGAN discriminator, which determines whether a local input image patch is real or fake. The PatchGAN discriminator(12) can alleviate the problem of the generator tending to exaggerate the characteristics of the data to deceive the discriminator. Layers:

First layer: Convolution with 64 filters, 4x4 kernel size, stride 2, and leaky ReLU activation. Second layer: Convolution with 128 filters, 4x4 kernel size, stride 2, instance normalization, and leaky ReLU activation. Third layer: Convolution with 256 filters, 4x4 kernel size, stride 2, instance normalization, and leaky ReLU activation. Fourth layer: Convolution with 512 filters, 4x4 kernel size, stride 1, instance normalization, and leaky ReLU activation. Final layer: Convolution with 1 filter, 4x4 kernel size, stride 1, for real/fake classification.

3.2.1.3 VGG context extractor

Considering limitations stem from the lack of input images, a perceptual loss is adopted, a combination of style and content loss inspired by (1), that enhances the visual quality by minimizing the feature distance between a target image and a stylecontent reference. to calculate it, a pre-trained VGG-16 network has been selected as the perceptual encoder, and it follows the same combination of the extract layers as the prior work (1), which is relu_2, relu. Style distance is gained between the input image and the reconstructed image, and the content distance is calculated between the input image and the generated target domain image.

3.2.1.4 Hyperparameters

The architecture and hyperparameters, such as the number of layers in the discriminator, were initially aligned with the baseline model. However, during experimentation, adjustments were made based on the observed performance, particularly in response to the loss functions and generated outputs. When the discriminator showed signs of dominance—where it consistently outperformed the generator, leading to training instability—its complexity was reduced by decreasing the number of filters or layers. This adjustment helped restore balance in adversarial training, ensuring that neither the generator nor the discriminator was overly dominant.

Similarly, the generator's filters and residual blocks were fine-tuned to enhance its ability to produce detailed and high-quality images. The filters in the first convolutional layer were increased when generated images lacked detail, and additional residual blocks were introduced to better capture complex patterns. However, these changes were carefully moderated to prevent overfitting and excessive training time. The number of shared blocks in the generator's centre module was also adjusted to improve feature sharing across tasks, enhancing generalization without sacrificing task-specific performance. These adjustments were iteratively tested to achieve better image quality and overall model stability.

3.2.2 Loss Functions

1. Cycle Consistency Loss

   It ensures that an image translated to a target domain and then back to the original domain remains unchanged. By enforcing cycle consistency, the CycleGAN framework prevents generators from excessive hallucinations and mode collapse, both of which will cause unnecessary loss of information and thus increase cycle consistency loss. Cycle consistency loss is focused on pixel loss hence, by measuring the difference between the reconstructed image's features and the original image extracted from an intermediate generator layer, feature loss is obtained. with this loss and pixel loss to alleviate enforcement of cycle consistency, weight is added as following equations :

   Initially Focus on detailed textures and fine details and the more epochs, the more focus on structural consistency and high-level features. It aims to balance between precise details and high-level feature preservation.

2. Latent Loss & Forward Loss

   It is computed between the original encoded features and the re-encoded features of the reconstructed images. It constrains the generator from producing latent features that are consistent with the original ones, promoting stability in the training process. This loss helps constrain the generator from producing latent features that are not drastically different from the original ones, thereby enhancing the stability and consistency of the model.

   On the other hand, forward loss ensures that when an image A is mapped to another domain B, the result B should as close as possible to A. This encourages the network to produce accurate domain translations. Focuses on the accuracy of domain translation, ensuring the output image closely resembles the original in terms of structure and content.

3. Identity Loss

   It is computed between the original encoded features and the re-encoded features of the reconstructed images. It constrains the generator from producing latent features that are consistent with the original ones, promoting stability in the training process. This loss helps constrain the generator from producing latent features that are not drastically different from the original ones, thereby enhancing the stability and consistency of the model.

4. Perceptual loss

   When a pretrained network exists, its early layers tend to view images similarly to the human perceptual process. In other words, the early layers transform the image into a perceptual space that is similar to our own. Given an input image x, pass it through an image transform like vgg net to extract the output y. Pass y and the content target y_c through to get the content loss, and pass y and the style target y_s through to calculate the style loss. Given the transformed image Y and the style target Ys, put each into the loss network to extract features and concatenate them. The result is a set of feature maps in the form of cubic tensors. At this time, gram matrices are matrices that indicate the degree of association between the horizontal and vertical axis channels. As a result, perceptual loss defines feature reconstruction loss and style reconstruction loss, which is a loss that tries to preserve the style and context of the image.

Final objective Our final objective is as follows. Figure3.3 illustrates our minimax learning objective. :

3.2.3 Self attention and Feature Extraction

Given the fact that we have only few dataset regarding ground refuelling system, extracting features method are employed to compare their effectiveness with other. There are several studies adopted segmentation or feature extraction to improve and compensate GAN's drawback and result. (23) Among these studies, self-attention and vgg19 net are employed to obtain pre trained model.

$$\mathcal{L}_{\text{total}}(G, F, D_A, D_B) = \mathcal{L}_{\text{Adversial}} + \lambda \cdot \mathcal{L}_{\text{cycle-consistency}} + \mathcal{L}_{\text{perceptual}} + \mathcal{L}_{\text{identity}}$$

Where:

- $\mathcal{L}_{\text{perceptual}}$ is a combination of content loss and style loss.
- $\mathcal{L}_{\text{identity}}$ encourages the generators to preserve the identity when images are already in the correct domain
- λ is the weight for the cycle-consistent loss.

Figure 3.3: Final objective

Self attention is aimed to select the information that is more critical to the current task goal from the global information, so It can make Self attention allows the model to capture long-range dependencies and relationships between different regions of the images, expecting it bring several effects on the model's performance,

The discriminator can focus on both local and global features of the image. Instead of just focusing on local texture details, the self attention module allows the discriminator to consider relationships between distant parts of the image. This helps in better distinguishing between real and fake images, especially when global structure and coherence are important. Also, the generator is incentivized to produce images with finer details and textures that are coherent across the entire image. Since the discriminator becomes more sensitive to inconsistencies over larger spatial areas due to the self-attention mechanism, the generator needs to match these expectations by generating more realistic textures and patterns. Overall, the generator can learn to apply these complex transformations more effectively, leading to more realistic and plausible domain translations. plus, the discriminator has a more global perspective, which can potentially stabilise the adversarial training process. The generator receives more informative gradients that consider both local and global aspects of the image, which can help in avoiding mode collapse and other common issues in GAN training.

Local features refer to the fine-grained, small scale details within an image such as texture, colour and edges. Regarding weather-cue, cloud texture, snowflakes on the ground can be considered local features.

3.2.4 Training Process

In the training process of ComboGAN, both the discriminator D and the generator G are optimized through a min-max game. Similar to CycleGAN, ComboGAN utilizes the same fundamental losses; however, the challenge arises in extending these to multiple domains, as the generator's cyclic training and the discriminator's true/false-pair training are primarily designed for two domains. ComboGAN addresses this by focusing on two of its n domains at a time during each training iteration.

To ensure uniform training across all domain pairs, the training process involves randomly selecting two domains per iteration, thereby eventually covering all possible domain pairs uniformly. This ensures that each domain is chosen for training an equal number of times, akin to the training process in CycleGAN. The discriminators' training procedure mirrors that of CycleGAN, where after each iteration involving a pair of generators, the corresponding discriminators undergo a training update.

During the mapping from one domain to another, the discriminator D updates its parameters by maximizing the adversarial loss objective. This adversarial optimization enables the generator to learn the real data distribution, ultimately allowing it to generate realistic synthetic images.

Discriminator and Generator Losses were used to monitor GAN training.

1. Discriminator Loss(D_loss) Measures how well the discriminator distinguishes between real and fake samples. Tracking this loss is important to ensure the discriminator doesn't become too strong or too weak.

2. Generator Loss (G_loss) The generator is fooling the discriminator. Monitoring this helps to see if the generator is improving over time.

3. Label smoothing (optional) Adding a smoothing factor to the discriminator's loss makes the training process more stable, reduces overfitting, and prevents the discriminator from overpowering the generator too quickly. It effectively introduces a form of regularization into the discriminator's learning process, which can lead to a more balanced and effective training of GANs.

   Label smoothing can help stabilize the training of the discriminator. By avoiding labels that are exactly 0 or 1, the discriminator is discouraged from becoming overly confident in its predictions. This can help prevent the discriminator from overpowering the generator too quickly and encourage a more balanced training process. Softer Decision Boundaries, making it slightly harder for the discriminator to distinguish them perfectly. This may encourage the generator to produce higher-quality fake data to fool the discriminator. The training is considered to have reached maturity when both reach Nash equilibrium, that is when the generator and discriminator promote each other to reach an optimal adversarial state.

Chapter 4

Experiment design

The primary objective of our experiments is to generate high-quality images of the aircraft ground refuelling system under various weather conditions. This goal is achieved by finding the optimal model based on ComboGAN, fine-tuning different hyperparameters and experimenting with various loss functions to enhance the visual realism and accuracy of the generated images. By systematically evaluating these models, it is aimed to identify the best-performing configuration that balances image quality and computational cost, thereby contributing to the advancement of automatic aircraft refuelling systems in diverse weather scenarios.

4.1 Experiment Environment

4.1.1 Hardware and Software

In order to conduct these experiments, we utilized a computational setup, leveraging PyTorch 3.0 with CUDA for efficient GPU acceleration on crescent 2, a virtual environment managed by Cranfield University. Both Linux and an NVIDIA GPU are prerequisites for the setup. The hardware configuration included NVIDIA GPUs (T4 or V100) supported by a high RAM capacity. This configuration ensures expedited training and testing processes, thereby enabling efficient handling of large volumes of image data and intricate computations intrinsic to the ComboGAN architecture.

4.1.2 Discussion of the Hyperparameters

Table 4.1: Basic Hyperparameter Settings

Hyperparameter	**Value**
Number of Generator Filters (ngf)	64
Number of Discriminator Filters (ndf)	64
Number of Residual Blocks in Generator ($netG\_n\_blocks$)	4
Learning Rate (lr)	0.002
Number of Layers in Discriminator ($netD\_n\_layers$)	9

The basic hyperparameters used in the experiment, including the number of filters in the generator and discriminator as well as the number of layers, were configured to match those of the baseline model. This decision was made to ensure a fair comparison and to isolate the effects of other modifications made during the experiment. Specifically, the architecture's depth and filter size are critical in determining the model's capacity to learn complex patterns, and maintaining consistency with the baseline model allowed for a clearer assessment of the impact of any adjustments.

The initial learning rate was set to 0.002, with a gradual decay applied after 100 epochs. This approach was chosen to strike a balance between convergence speed and stability. The relatively higher initial learning rate helps the model make rapid progress during the early stages of training, while the gradual decay helps fine-tune the model by preventing overshooting as it approaches convergence.

To enhance the stability and robustness of the training process, the Least Squares Error (LSE) loss function was adopted, following the approach used in CycleGAN. LSE was selected instead of the traditional sigmoid cross-entropy loss due to its advantages in addressing the diminishing gradients problem. Unlike the sigmoid cross-entropy loss, which tends to saturate when the input x is relatively large—leading to vanishing gradients—LSE maintains a consistent gradient throughout the training process. This characteristic ensures that the model continues to receive meaningful updates, even as the discriminator becomes more confident.

Moreover, because the least squares loss function is minimized only at a specific point, it provides a smoother and more stable training dynamic, reducing the risk of mode collapse—a common issue in GAN training. This makes LSE particularly suitable for tasks where precise and stable convergence is crucial, such as generating aircraft ground refueling images in various weather conditions, as it allows the generator to produce higher quality outputs without the instability that might arise from other loss functions.

The choice of LSE aligns with the methodology outlined by Mao et al(24). in their work on Least Squares GANs, which demonstrated the effectiveness of this approach in maintaining gradient flow and achieving stable convergence.

Table 4.2: Hyperparameters of loss functions Settings

Hyperparameter	**Value**
λ_{cycle}	10
$\lambda_{identity}$	0.1
$\lambda_{perceptual_style}$	0.5
$\lambda_{perceptual_content}$	0.5
λ	0

For optimal translation performance, it is essential to adhere to the hyperparameter configurations specified in Table 5.2. This table delineates the most effective settings for loss factors such as perceptual loss, gradient penalty magnitudes, and cycle consistency loss for each translation task.

It's important to note that due to the volatility of GANs training, some of the values in Table4.2 may undergo slight changes due to random fluctuations.

4.1.3 Ablation Study

To assess the actual effectiveness of each factor, we conducted an ablation study in various environments. All experiments were carried out in identical environments, except for the factor being tested.

1. latent loss & forward loss
2. perceptual loss
3. identity loss

4.1.4 Evaluation Metrics

4.1.4.1 Qualitative Evaluation

Qualitative evaluation was conducted to assess the visual quality of the generated images from both the AGR dataset, which includes images of aircraft ground refuelling systems, and the BDD100K Road dataset. The primary focus of this evaluation was on the AGR dataset, as the ultimate goal of this study is to generate realistic images of ground refuelling systems under various weather conditions. However, the BDD100K Road dataset was also evaluated to validate the broader applicability and performance of the model across different types of imagery.

The qualitative assessment involved a detailed visual inspection of the generated images, focusing on key criteria such as realism, consistency with the target weather conditions, and overall image coherence. Specifically, the following aspects were considered:

1. Realism The extent to which the generated images closely resembled real-world scenes, with particular attention to the accuracy of objects' shapes, textures, and spatial relationships within the scene.
2. Consistency with Target Weather Conditions The ability of the model to accurately reproduce the desired weather conditions, such as fog, rain, or clear skies, in the generated images. This was critical in evaluating how well the model could apply these conditions to the specific context of ground refueling systems.
3. Image Coherence and Detail The evaluation also included an analysis of the internal consistency of the images, ensuring that weather effects were uniformly applied across the scene and that no artifacts or inconsistencies were present. Additionally, the preservation of fine details, such as the clarity of aircraft and ground equipment under different weather conditions, was a key focus.

Through this qualitative evaluation, the visual fidelity of the generated AGR images was scrutinized, helping to identify strengths and potential areas for improvement in the model. The BDD100K Road dataset images were similarly inspected to confirm that the model's capabilities were not confined to a specific dataset, thereby supporting the robustness of the approach.

4.1.4.2 Quantitative Evaluation

The Inception Score (IS) is a widely used method for evaluating the performance of GANs among generative models. It is based on two key metrics: sharpness and diversity. Before delving into these metrics, it's essential to understand the concept of entropy. In the context of probability, entropy measures the degree of uncertainty or disorder associated with a random variable. Specifically, when considering the conditional distribution p(y—x), high entropy indicates that there are many possible values of y given x, making y difficult to predict. Conversely, low entropy suggests that y is highly predictable for x.

1. Sharpness (S) Sharpness refers to the confidence with which a classifier predicts a label y for a given image x. For example, consider an image of the digit '4' generated by a GAN trained on the MNIST dataset. If a digit recognizer classifies this image as a '4' with high confidence, the image is considered to be of good quality. Mathematically, sharpness is related to the entropy of the classifier's predictive distribution p(yx). A large sharpness corresponds to low entropy in p(yx), indicating that the classifier is making confident predictions. This can be represented as:

   $\mathrm{S(x)} = \exp\left(E_x\left[\mathrm{KL}\left(p(y|x) \parallel p(y)\right)\right]\right)$

2. Diversity (D) In addition to sharpness, it is crucial that the generated images are diverse. Diversity is evaluated by examining the marginal distribution p(y), which reflects the variety of outputs produced by the generator. High diversity ensures that the model is not generating the same image repeatedly but is instead producing a wide range of outputs. This is important for the overall quality of the GAN, as it reflects the model's ability to generate varied samples. Diversity can be expressed as the entropy of the marginal distribution p(y):

   $\mathrm{D(x)} = \mathrm{E}_x\left[\mathrm{H}(p(y))\right]$

3. Inception Score (IS) The Inception Score is then calculated by combining sharpness and diversity. Specifically, it is the exponential of the Kullback-Leibler divergence between the conditional label distribution p(yx) and the marginal label distribution p(y), averaged over all generated images:

   $\mathrm{IS} = \exp\left(E_x\left[\mathrm{KL}\left(p(y|x) \parallel p(y)\right)\right]\right)$

   A higher Inception Score indicates that the generated images are both sharp (i.e., the classifier can predict them with high confidence) and diverse (i.e., the generated images cover a wide range of classes). The IS is widely regarded as being strongly correlated with human judgment, making it a reliable metric for assessing GAN performance. Typically, if a classifier is not available, the Inception model pre-trained on ImageNet is used, which is why the metric is named the Inception Score.

4.2 Comparison experiments

Baseline Models: In this study, we systematically evaluated various configurations of hidden layers, learning rates, and loss functions to identify the optimal values that yield both quantitative and qualitative improvements in model performance. Consequently, the

final model was fine-tuned using these hyperparameters through feature extraction, with different settings manually selected for model testing.

Layers: We initially trained both the generator and discriminator using an equal number of hidden layers, analyzing the impact on the quality and loss graphs of each. Different configurations were tested across various training sessions. It was generally observed that increasing the number of layers in the discriminator relative to the generator led to faster learning, as the generator often outpaces the discriminator. This imbalance can be problematic, as the generator should ideally produce increasingly convincing images to challenge the discriminator, which must then accurately distinguish between real and generated images. Therefore, we prioritized adjusting the number of layers and learning rate for the discriminator, while largely maintaining the generator's hyperparameters. Although the difference was minimal, as illustrated in Table 4.1 we found that using 64 layers for both the generator and discriminator provided greater stability in our experiments compared to configurations with a higher number of layers in the discriminator

Table 4.3: IS and FID values for ablation experiments

model	**FID**
comboGAN	125
comboGAN + self-attention	124
comboGAN + perceptual loss	120

The images below discusses the effects of identity loss on images. When an image is regenerated, it is almost identical to the original image, even if the fake image is blurred or distorted. With identity loss, perceptual loss xx.. but there may be limitations with pre-trained image transformation models, leading to insufficient segmentation.

Chapter 5

Results and Discussion

We first describe the results of the final model, then analyse the factors that were and were not used in the final model through an ablation experiment, and finally compare it to the original baseline model, combogan, using qualitative and quantitative metrics.

5.1 Experiment Results

Table 5.1: FID scores on the BDD and AGR datasets. The model was trained on 300-400 images from the BDD100K dataset and 30-50 images from the AGR dataset for each domain. Lower is better.

model	**BDD 100k**	**AGR dataset**
Sunny2Cloudy	86.26	102.87
Snowy2Sunny	102.2	103
Cloudy2Snwowy	110.25	120.92
Rainy2Sunny	134.87	141.7

5.1.1 Comparison to baselines

Figure5.4 compares the baseline model and our final approach with perceptual loss and weighting cycle consistency. In both experiments, all the hyperparameters are set up identically with a decay learning rate as linearly decaying from the initial value to 0 for another 100 iterations to the end.

It shows that, by adapting those changes, the result has been improved with fewer artifacts than ComboGAN and much more natural weather cues. Also, our final model performed reasonably adequate in converting images across the four different weather domains: sunny, cloudy, snowy, and foggy. The generated images demonstrate a plausible degree of realism, particularly in their ability to accurately represent various weather conditions. For instance, sunny images exhibit strong lighting and blue sky, while cloudy images show a cloud texture in sky.

However, a closer inspection reveals that some generated images contain artifacts that suggest the model is encoding unnecessary information from the training process. These

Loss	FID	IS
ComboGAN	125	4.92
ComboGAN + consistency weight decay + perceptual loss	118	5.23

Table 5.2: The FID score and IS on different approached model. For FID, Lower is better and for IS, Higher is better. Last row is final approach

artifacts are often semantically meaningful, such as the inversion of clouds in cloudy images, which can occasionally result in unrealistic sky patterns. In other cases, the model uses simpler strategies, like color inversion, to toggle between different weather conditions, which can lead to inconsistencies in the overall color scheme.

Despite these issues, the model generally maintains coherence and detail across the various weather conditions. Snowy scenes, for example, retain the structure and texture of snow-covered surfaces, while rainy images depict wet roads and reduced brightness. The preservation of fine details, such as reflections on wet surfaces or the texture of snow, contributes to the overall quality of the generated images.

The model's performance on the AGR dataset further supports these findings. While the generated images are largely successful in replicating the desired weather conditions, some instances show minor inconsistencies, particularly in more complex scenarios like fog, where subtle gradations of light and shadow are crucial. These results suggest that while it is effective in generating weather-appropriate images, further refinement is needed to eliminate artifacts and ensure greater consistency across all weather conditions.

In Table5.2, we also compare with quantitative measures method to baselines. On both quantitative methods across datasets, FID and IS slightly outperform the baseline model.

5.1.2 Ablation study

5.1.2.1 Identity Regularization

Experiments are conducted with below 3 different backgrounds: ComboGAN without any regularization, ComboGAN with only perceptual loss as in original Perceptual loss GAN setting and ComboGAN with perceptual loss and weight decay cycle consistency loss. All models are trained for 100n epochs on single Nvidia V100 8GB GPU with a batch size of 1. For loss factors, the same setting as the original ComboGAN was used, with $(\lambda\_) = 0.1, (\lambda\_) = 10, and for experiments with identity loss and initial weight for the cycle consistency loss which is gra$
$0 were reset By adding identity loss, it results in better resolution and inclined to show similarity with original image, a$

5.1.2.2 Cycle Consistency loss

Baseline models tend to catch colour because cycle consistency loss and GAN loss are pretty good at guiding generators to output image with correct colours. However, at the same time, it led to hallucination as shown in the very right images on Figure5.5.

In (13), they proved that the two generators, rather than focusing on producing realistic images, may instead learn a color inversion mapping that allows them to collectively minimize the cycle consistency loss. This behavior can trap the generators in local minima, from which they are unlikely to escape due to the constraints imposed by the cycle

consistency loss. Consequently, enforcing cycle consistency on cycles involving unrealistic generated images can impede the overall training process.

To address this issue, it is adopted to enforce a weaker form of cycle consistency by incorporating an L1 loss on the CNN features extracted by the corresponding discriminator as identical to this paper. The discriminator, ideally having learned meaningful features for the target image domain, helps guide the generator toward more realistic outputs. Specifically, the modified cycle consistency loss for one translation direction is defined as a linear combination of CNN feature-level and pixel-level consistency losses. The parameter gamma governs the balance between the discriminator's CNN feature-level loss and the pixel-level loss. Initially, gamma is set to a low value since the discriminator's features are not well-developed at the start of training. It gradually increases linearly to a value close to, but not equal to, 1 to maintain some degree of pixel-level consistency. This balance helps prevent excessive hallucination of background and unrelated objects in the images. The results of introducing this loss are shown in Table**??**

5.1.2.3 Stabilization of training

An experiment was carried out to stabilize the training process by implementing spectral normalization, introduced in (25) Spectral normalization is a technique that maintains stable GAN training by defining a Lipschitz constant, k, ensuring that the difference between two distributions does not increase beyond a factor of k. This is crucial for consistent training of the discriminator.

Recent research has highlighted the significant impact of conditioning the generator on GAN performance. Building on this, Miyato et al.(25) applied spectral normalization to the GAN generator and observed its effectiveness in improving training stability. The incorporation of spectral normalization into the generator can prevent parameter magnitudes from growing excessively and avoid abnormal gradients.

Implementing spectral normalization for both the generator and the discriminator has additional advantages. It can reduce the number of updates needed for the discriminator compared to the generator, thus lowering the computational cost for training. Additionally, it contributes to overall training stability.

Based on this background, spectral normalization for both the generator and the discriminator was integrated into the model. However, in the context of the specific dataset and model, significant improvements were not observed, and this approach was not extensively explored in the analysis.

To ensure balanced training, close monitoring of the GAN loss was conducted throughout the process. If one side converged prematurely, effective training could not proceed. Several methods were explored to strengthen the discriminator and compared for their outcomes.

Among these methods, label smoothing emerged as the most effective. This technique slowed down the training convergence of the discriminator over extended training epochs. A comparison with the standard model demonstrated that learning progress was indeed slower when label smoothing was applied.

5.1.2.4 Perceptive loss

Models utilizing perceptual loss demonstrated reduced inconsistencies in overall composition and mitigated issues such as unnatural color shifts. However, in scenarios involving rainy weather images, these models often failed to adequately capture rain features. Furthermore, during translation to sunny weather images, the models frequently left rain artifacts intact while merely altering the sky color to blue. In contrast, models implementing the contrastive loss approach, as introduced in the CUT (Contrastive Unpaired Translation) paper, yielded more plausible results for rainy weather scenarios. Although these images exhibited slightly reduced sharpness, they better conveyed the intended weather conditions. The contrastive loss functions by aligning features from corresponding patches across the input and translated images, thereby encouraging the preservation of key content from the original image.

5.1.2.5 Self attention

In this ablation study, we integrated a self-attention layer into the discriminator, inspired by recent advancements in attention mechanisms that demonstrate the potential of leveraging peripheral information that aligns with object contours rather than merely focusing on local, fixed regions. Self-attention mechanisms are designed to utilize cues from all feature locations, thereby enhancing the detail and coherence of feature representations.

The primary objective behind adding a self-attention layer was to enhance the discriminator's capabilities beyond mere binary classification. Specifically, we aimed to enable the discriminator to enforce consistency in feature representations across distant regions within an image. Traditional convolutional networks (ConvNets) are inherently limited by their local receptive fields, capturing only short-range dependencies within the image. While convolutional operations can gather distant features through processes like global average pooling, this is typically achieved only after several layers, making it difficult for smaller models to represent long-range dependencies effectively.

The reliance on convolutional layers within GANs, while powerful, imposes a constraint due to the local nature of the receptive fields. To capture dependencies between distant parts of an image, ConvNets require multiple layers, leading to increased model complexity. Expanding the convolutional kernel to address this issue would be computationally expensive and would undermine the statistical efficiency that local convolutions provide.

The introduction of self-attention was motivated by the need to balance computational efficiency with the capacity to capture long-range dependencies. By assigning weights to features at every location, self-attention mechanisms compute the response at a specific location while considering the entire feature map, all with a relatively lower computational cost.

However, despite these theoretical advantages, our empirical results did not exhibit the expected improvements. The top image in the figure5.8 is the result of introducing a self-attention layer to generate a sunny weather image from each of the foggy, rainy, and cloudy images. These are the images obtained by our model, Specifically, the generated images—intended to depict sunny weather from foggy, rainy, and cloudy conditions—showed better feature localization but suffered from a slight decrease in overall image accuracy and sharpness. This suggests that while self-attention can enhance the

capture of specific features, it may introduce trade-offs in other aspects of image quality. Further investigation is needed to optimize the integration of self-attention within the discriminator to fully harness its potential benefits.

5.2 Testing Visualization

Due to the nature of GAN, which is a min max game of generator and discriminator, the losses of the two do not converge, but sometimes the generator loss is dominated and sometimes the discriminator loss is dominated, and it is not easy to judge or stop the training by the loss function, unlike the general NN model, because it does not converge. We calculated the FID score at every epoch to track whether the training is going well. to determine when to stop the training, FID score has tracked with our final model. it is trained with 3 domains and after 300 epoch, FID seems to not improve considering cost, train was stopped. 5.9

Figure 5.1: Result with AGR dataset

Figure 5.2: Result with BDD 100k Road dataset

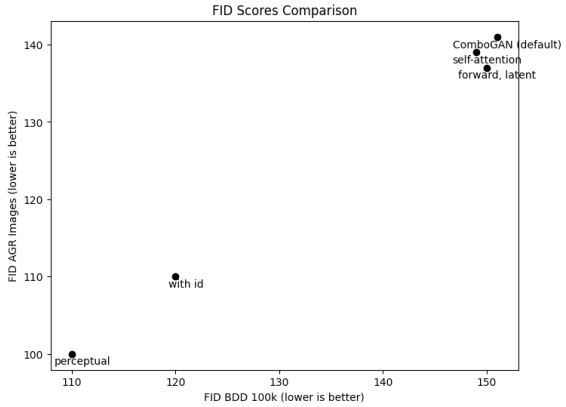


Figure 5.3: Comparsion with other loss regularization. We plot the FIDs on both BDD datasets and AGR dataset. A few approach which show at least improvement has shown on this plot. Some approach which hurt performance are excluded

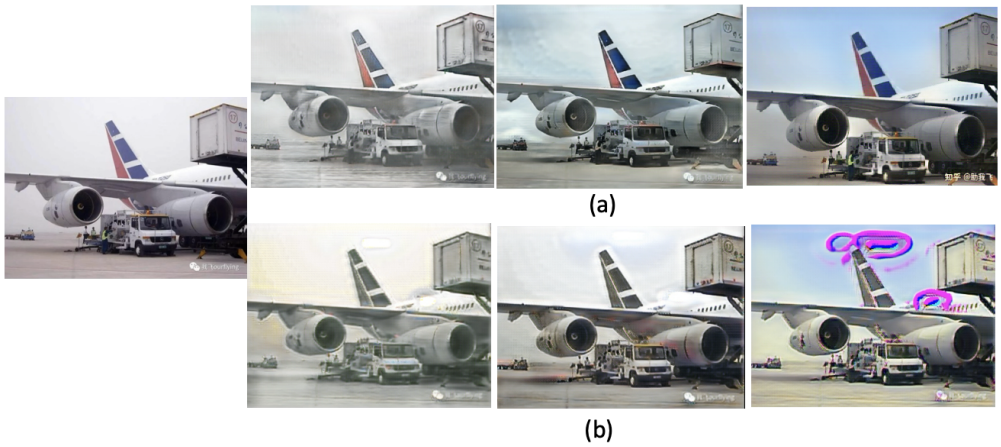


Figure 5.4: The comparison with ComboGAN and final approach on AGR validation dataset. The left image indicates input images, others are results. From left, it is a generated snowy, cloudy and sunny from the original foggy image (a) ComboGAN + perceptual loss + cycle - consistent with weight (b) Baseline ComboGAN model

Figure 5.5: Caption

Figure 5.6: Result of the model with final objectives in generating weather image from rainy image

Figure 5.7: Result of the model with contrastive loss approach in generating weather image from rainy image

Figure 5.8: (top) self attention results (bottom) our model results

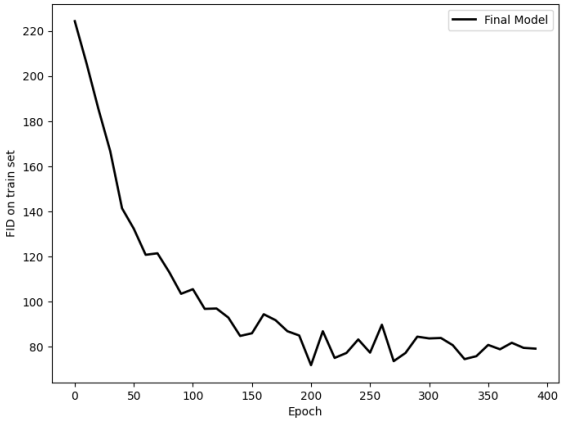


Figure 5.9: FID score has tracked with ComboGAN with identity loss. it is trained with 3 domains

Chapter 6

Conclusion

ComboGAN was initially proposed as an extension of CycleGAN to handle multiple domains within a single framework, similar to an autoencoder architecture. The primary advantage of ComboGAN lies in its efficiency; it reduces the need for multiple training sessions across different domains. However, the model's structure, which splits the generator into encoder and decoder halves, introduces complexities in training and tuning, often leading to performance degradation.

To address these challenges, various enhancements were introduced to improve ComboGAN's effectiveness. Among the techniques tested, the most successful modifications were selected for this project. Perceptual loss was incorporated to prioritize feature extraction over pixel-level accuracy, which helped in producing more realistic images. Additionally, the introduction of a weight decay factor into the cycle consistency loss mitigated some of the inherent limitations of CycleGAN, leading to more plausible results.

However, balancing the different types of losses within the model presents a potential trade-off. In this project, the initial loss weights were either adopted from previous research or chosen randomly, which may not fully optimize the model's performance. Future work could focus on refining these weights based on specific objectives, such as placing more emphasis on aspects that are expected to yield better results.

The inherent variability of GAN models means that even with the same dataset and architecture, output quality can be inconsistent. In this context, shared layers between the encoder and decoder might cause conflicts; for example, the encoder may prioritize high-level feature extraction while the decoder requires finer details, potentially compromising the quality of generated images. Another potential improvement could involve calculating perceptual loss using a self-trained image classification model rather than relying on a pre-trained model like VGG-16, which might better align with the specific domains of this project.

References

1. Johnson J, Alahi A, Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution. European Conference on Computer Vision (ECCV). vol. 9906; 2016. p. 694-711.

2. D W Jaw SCH, Kuo SY. Desnowgan: An efficient single image snow removal framework using cross-resolution lateral connection and fans. EEE Transactions on Circuits and Systems for Video Technology. 2020.

3. D Engin AG Ekenel HK. Cycle-dehaze: Enhanced cyclegan for single image dehazing. IEEE Conference on Computer Vision and Pattern Recognition. 2018.

4. S Yildirim, Z Rana and G Tang. Autonomous Ground Refuelling Approach for Civil Aircrafts using Computer Vision and Robotics; 2021. Available at: `https://ieeexplore.ieee.org/document/9594312`.

5. Lotter W, Kreiman G, Cox D. Unsupervised Learning of Visual Structure using Predictive Generative Networks; 2015.

6. Kingma DP, Welling M. Auto-Encoding Variational Bayes; 2022. Available at: `https://arxiv.org/abs/1312.6114`.

7. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al.. Generative Adversarial Networks; 2014. Available at: `https://arxiv.org/abs/1406.2661`.

8. Goodfellow I. NIPS 2016 Tutorial: Generative Adversarial Networks; 2017. Available at: `https://arxiv.org/abs/1701.00160`.

9. Choi Y, Choi M, Kim M, Ha JW, Kim S, Choo J. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation; 2018. Available at: `https://arxiv.org/abs/1711.09020`.

10. Zhu JY, Park T, Isola P, Efros AA. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks; 2020. Available at: `https://arxiv.org/abs/1703.10593`.

11. Anoosheh A, Agustsson E, Timofte R, Gool LV. ComboGAN: Unrestrained Scalability for Image Domain Translation; 2017. Available at: `https://arxiv.org/abs/1712.06909`.

12. Demir U, Unal G. Patch-Based Image Inpainting with Generative Adversarial Networks; 2018. Available at: https://arxiv.org/abs/1803.07422.

13. Wang T, Lin Y. CycleGAN with Better Cycles; 2024. Available at: https://arxiv.org/abs/2408.15374.

14. Xu X, Liu J, Liu W. MCMI: Multi-Cycle Image Translation with Mutual Information Constraints; 2020.

15. Yunjey Choi MKJWHSK Minje Choi, Choo J. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

16. Arjovsky M, Chintala S, Bottou L. Wasserstein GAN; 2017.

17. Zhang R, Pfister T, Li J. Harmonic unpaired image-to-image translation; 2019.

18. Name(s) A. Title of the Paper on Contrastive Learning for Unpaired Image-to-Image Translation. Journal Name. Year.

19. Name(s) A. Title of the Paper on Geometry-Consistent Generative Adversarial Network (GcGAN). Journal Name. Year.

20. Name(s) A. Title of the Paper on DistanceGAN. Journal Name. Year.

21. Zhang H, Goodfellow I, Metaxas D, Odena A. Self-Attention Generative Adversarial Networks; 2019. Available at: https://arxiv.org/abs/1805.08318.

22. Yu F, Xian W, Chen Y, Liu F, Liao M, Shen W, et al.. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning; 2020. https://bdd-data.berkeley.edu/. Accessed: 2024-08-30.

23. Mondal AK, Agarwal A, Dolz J, Desrosiers C. Revisiting CycleGAN for semi-supervised segmentation; 2019. Available at: https://arxiv.org/abs/1908.11569.

24. Bousquet MLKKMMO, Gelly S. Are GANs Created Equal? A Large-Scale Study; 2018.

25. Miyato T, Kataoka T, Koyama M, Yoshida Y. Spectral Normalization for Generative Adversarial Networks; 2018. Available at: https://arxiv.org/abs/1802.05957.