KU LEUVEN

Comic Art Generation using GANs

ir. Marnix Verduyn

Thesis submitted for the degree of
Master of Science in Artificial
Intelligence, option Engineering and
Computer Science

Thesis supervisor:
prof. dr. ir. Luc De Raedt

Assessors:
prof. dr. Tim Van de Cruys
ir. Thomas Winters

Mentor:
ir. Thomas Winters

Academic year 2021 – 2022

Preface

I thank my promoter prof. dr. ir. Luc De Raedt, who showed a lot of interest in the research of an AI application on the borderline between science and art.

I thank my daily supervisor ir. Thomas Winters for sharing his extensive knowledge about creative AI, for his contagious enthusiasm, his motivation, and for his good advice.

Also a big thank you to my wife Lies and my daughter Lus for the good vibes.

ir. Marnix Verduyn

Contents

Preface i

Abstract iv

List of Figures and Tables v

List of abreviations and symbols viii

Abreviations . . . viii

1 Introduction 1

1.1 Drawing Style Imitation. . . . 1
1.2 Kinky & Cosy . . . 2
1.3 GANs . . . 2
1.4 Structure . . . 3

2 Related Work 5

2.1 Comics Research in Computer Science . . . 5
2.2 Generative Adversarial Networks . . . 8
2.3 Conclusion . . . 17

3 Comic Art Generation using DCGAN 19

3.1 Dataset . . . 19
3.2 Architecture . . . 19
3.3 Training . . . 20
3.4 Evaluation . . . 22
3.5 Improvement by Adjusting the Dataset . . . 23
3.6 Higher resolution . . . 25
3.7 Conclusion . . . 26

4 Comic Art Generation using WGAN 27

4.1 Dataset . . . 27
4.2 Architecture . . . 27
4.3 Training . . . 28
4.4 Conclusion . . . 29

5 Comic Art Generation using StyleGAN2-ADA 31

5.1 Dataset . . . 31
5.2 Architecture . . . 31
5.3 Training . . . 32
5.4 Evaluation . . . 34

5.5 Hyperparameters . . . 36
5.6 The Latent Space . . . 37
5.7 Meaningful Interpolations . . . 39
5.8 Projection . . . 40
5.9 Network Blending . . . 41
5.10 The Pre-trained Model . . . 42
5.11 A Higher Resolution . . . 44
5.12 Transfer Learning on Comic Dataset . . . 45
5.13 Conclusion . . . 45

6 Conclusion **47**
6.1 Future Work . . . 47

A Experimental Set-up **51**
A.1 The Dataset of Kinky & Cosy . . . 51
A.2 Pre-trained StyleGAN2 Models . . . 54

Bibliography **55**

Abstract

This thesis arose from the search for a solution to a practical problem in the art world. If a well-known comic book author is no longer able or willing to continue working on his oeuvre, can a machine continue his work? The following pages describe an empirical study of the technical difficulties of comic art generation using GANs on an unstructured dataset of limited size. GANs have proven to be very powerful in creating real images since the internet is abundant with pictures uploaded by users. I show that the success of GANs with photographic images does translate only moderately into generating high-quality comics panels, because of the limited size and the diversity of the dataset at hand. In a comparison between DCGAN, WGAN, and StyleGAN2-ADA, the latter proves to be impressively well suited to modeling the underlying data structure. Although the synthesized drawings are not yet suitable for publication due to the presence of many imperfections, the results of this research are promising. It might be the first step in the development of an autonomous comic author.

List of Figures and Tables

List of Figures

1.1 A comparison between the original Gaston and an excellent imitation. ©Éditions Dupuis . . . 2
1.2 The newspaper strip Kinky & Cosy. ©Nix . . . 3

2.1 Generated anime faces by [Jin et al., 2017], the picture was taken from the paper. . . . 7
2.2 AnimeGAN [Chen et al., 2020] creates comic panels in manga style with neural style transfer. Picture ©Marnix Verduyn. Style transformation created with [Long, 2020] . . . 8
2.3 DCGAN Generator (picture taken from the paper [Radford et al., 2015]). 11
2.4 StyleGAN: a style-based generator with two networks f and g (picture taken from the paper [Karras et al., 2018]). . . . 13
2.5 StyleGAN2: a "demodulation" operation is applied to the weights associated with each convolution layer (picture taken from the paper [Karras et al., 2019]). . . . 15
2.6 A real image (upper-left) and a projected image (upper-right) with an FFHQ pre-trained StyleGAN2. A style-transferred image (lower-left) between the real image and the comic character (lower-right). Real image: ©Cyrus Pâques . . . 16

3.1 Training data: stills from the animation movies, cropped left and right. 20
3.2 Loss versus training iterations for the DCGAN. . . . 21
3.3 Generated images with DCGAN trained on cropped stills (size 64×64 pixels). . . . 22
3.4 New training data: selected stills from the animation movies, focusing on the main characters Kinky & Cosy. . . . 23
3.5 Generated images with DCGAN trained on images focussed on Kinky & Cosy (size 64×64 pixels). . . . 24
3.6 Linear interpolation between latent points for DCGAN. . . . 25

4.1 Loss versus training iterations for the WGAN. . . . 28
4.2 Generated images with WGAN trained on cropped stills (size 32×32 pixels). . . . 29

5.1 Clockwise : D-loss, G-loss, D sores for real images and FID for the dataset of general images. 33
5.2 Clockwise:D-loss, G-loss, D sores for real images and FID for the dataset of images focussed on Kinky & Cosy. 33
5.3 Generated images with StyleGAN2-ADA trained on general cropped stills (size 128×128 pixels) 34
5.4 Generated images with StyleGAN2-ADA trained on images focussed on Kinky & Cosy (size 128×128 pixels) 35
5.5 FID for different values of $\gamma = 0.5(green), 5(blue), 50(orange), 500(red)$. 37
5.6 The mode of G's output distribution. (size 128×128 pixels) 38
5.7 Images generated with the same seed, but with $\mathbf{z}$ sampled from a truncated normal with increasing magnitude (from upper-left image to lower-right ranging for 0 to 20 with step 0.1) 39
5.8 A circular walk through the latent space with random seed 67, truncation 0.5 and diameter 25 40
5.9 Example of a semantically meaningful direction in the latent space $\mathcal{W}$. Kinky (on the right) turns her head smoothly from $\frac{3}{4}$ left to $\frac{3}{4}$ right in this interpolation. 41
5.10 This walk in the latent space $\mathcal{W}$ is also a walk on the sidewalk for the Kinky & Cosy. It is a semantically meaningful direction in $\mathcal{W}$ because both characters stretch their legs as part of smooth walk animation. Additionally, Kinky (on the left) turns her head towards 'the camera' and smiles. 42
5.11 Projection of images in the latent space. Top: the source images, bottom: the targets. From left to right: a generated image, an image from the training set, and an image that is never seen by the model. 43
5.12 Generated images from blended models. Left: lower layers from FFHQ and higher layers from Kinky & Cosy with a split at resolution 128^2. Right: the reverse. 43
5.13 Fake images after 12kimg of transfer learning on model pre-trained on the FFHQ dataset. 44
5.14 Real images of Kinky & Cosy comic strips. 45
5.15 Fake images of Kinky & Cosy comic strips after transfer learning on the model pre-trained on the Kinky & Cosy animated cartoons. 46

A.1 In the design of the animation series, a turn of Cosy consists of a limited number of drawings. 52
A.2 A T-SNE plot of 2.000 consecutive samples from the dataset of images focusing on the main characters Kinky & Cosy. 53
A.3 A collection of pre-trained StyleGAN2 models trained on different datasets: Anime portraits, My Little Pony, Modern art, Human Faces, Abstract art. 54

List of Tables

A.1 The number of drawings in the design phase. 52

List of abreviations and symbols

Abreviations

MLP	Multilayer Perceptron
GAN	Generative Adversarial Network
DCGAN	Deep Convolutional GAN
WGAN	Wasserstein GAN
WGAN-GP	Wasserstein GAN with Gradient Penalty
PGGAN	Progressive Growing GAN
ADA	Adaptive Discriminator Augmentation
AdaIN	Adaptive Instance Normalization
R1	A regularization technique and gradient penalty for training GANs
PPL	Perceptual Path Length
LPIPS	Learned Perceptual Image Patch Similarity
VGG16	A very deep convolutional network for large-scale image recognition
SLERP	Spherical linear interpolation

Symbols

kimg	1.000 images in training GANs
fps	frames per second

Chapter 1

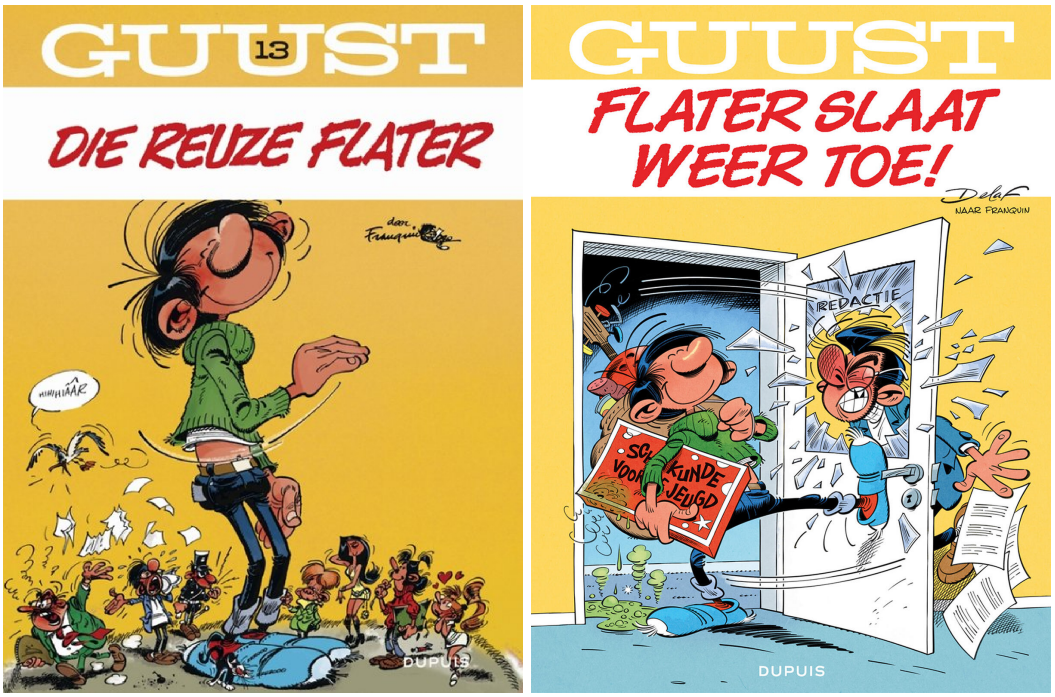
Introduction

1.1 Drawing Style Imitation.

In practice, there are only a limited number of books by a particular author. Let's take the comic series Gaston as an example. When the author Franquin died in 1997, he left a body of work of 19 albums. Every album counts 44 pages with an average of 12 panels per page. This means that there are approximately 10.000 panels of Gaston available. This is the dataset to learn the style of Franquin on the series Gaston. If the imitator were human, they would study these panels carefully to understand how Franquin synthesizes humans, animals, and the rest of the world in his baroque style. In the case of machines, we have GANs at our disposal.

This research question is very topical in the case of Gaston. Around this time, the publishing house Dupuis wants to launch the first new comic book of Gaston that was not drawn by Franquin. The artist Delaf was asked to do this. For three years Delaf studied Franquin's drawing style, copied it, practiced it, and finally arrived at an imitation that can be called astonishing (compare figures 1.1a and 1.1b). But it also took a tremendous amount of effort for this young author, as he reveals in an interview: "I decided to build a database. I patiently cut out all of Franquin's original plates, and I ended up with 10.000 small files that I tagged to easily find references to objects, characters, settings, or attitudes. This documentation work allowed me to have references available at all times to be able to interpret the scenes properly. For one plate, I can have up to 300 references on the side screen of my computer."[Detournay, 2022]

What we learn from this is that people understand the semantics of drawings. Objects, characters, and backgrounds may be represented distorted in a caricatured style, but for people, the meaning of the drawings is entirely clear. As we will see, that is a clear advantage over the work done by GANs. Even though developments in recent years have given us models that produce amazing results. Unconditional GANs do nothing less or more than model a data distribution and generate samples.



(a) Gaston by Franquin (1973) (b) Gaston by Delaf (2021)

Figure 1.1: A comparison between the original Gaston and an excellent imitation. ©Éditions Dupuis

1.2 Kinky & Cosy

This research question doesn't come completely out of the blue. I have been a comic book artist myself for over twenty years. I know the challenges and difficulties of the comic book creation process. For this thesis, I don't even have to violate any copyrights because I can use my own work. The series on which the research is being conducted is called Kinky & Cosy and is an international newspaper strip centered around two blonde twin girls with a red dress (see figure 1.2).

1.3 GANs

This thesis aims to answer whether Generative Adversarial Networks (GANs) can be used for the unconditional generation of comic panels. Recent advancements in GANs have increased the capability of image synthesis to hyper-realistic levels. But the performance of GAN models is almost always assessed on photographic images. In some exceptional cases, GAN models are trained on datasets of modern art paintings. The generated results easily look similar, since the source material was abstract. For comics nevertheless, the results should be visually meaningful and flawless. So to extend experimental knowledge of unconditional GAN performance into the domain of comics, an empirical analysis is conducted on the unconditioned generative

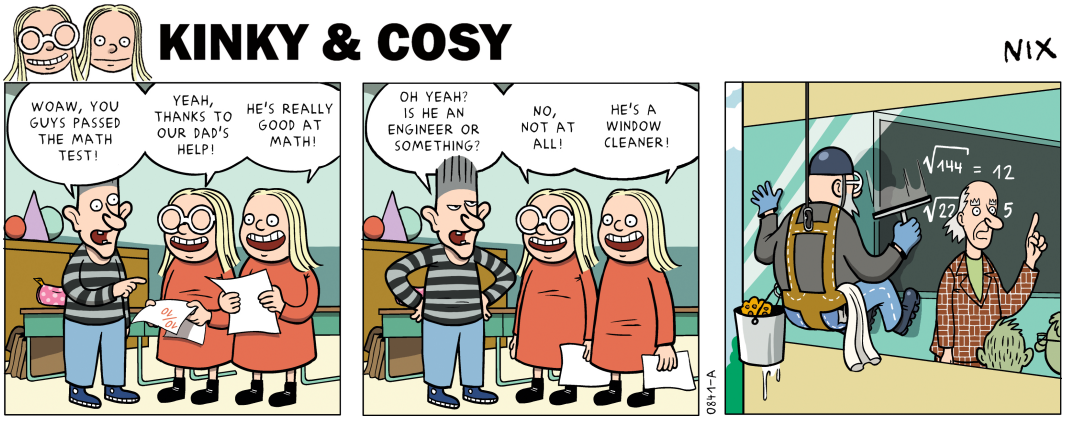


Figure 1.2: The newspaper strip Kinky & Cosy. ©Nix

performance of three state-of-the-art GAN architectures: Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and StyleGAN2-ADA. This thesis shows that the StyleGAN2-ADA implementation outperforms both the DCGAN and WGAN architectures on a dataset of Kinky & Cosy comic panels. Recently similar work has been done on other datasets like the Dilbert newspaper comic ([Morris, 2021] and [Proven-Bessel et al., 2021]). Dilbert is however characterized by simplicity, repetition, and similarity between the panels and as such a dataset that easily results in GAN-generated images that visually resemble the originals. With the dataset of Kinky & Cosy, we will see that more diversity leads to more non-sensical drawings. The results of the experiments in this thesis are evaluated by both metrics as FID, as well as by a visual evaluation performed by a small test audience.

1.4 Structure

This thesis is structured as follows. Chapter 2 discusses the related work. Chapters 3, 4 and 5 provide an overview of an empirical analysis study aimed at comparing results of the previously mentioned architectures on unconditional comic panel generation. Chapter 6 provides a conclusion of the research as well as suggestions for future work. Appendix A completes the thesis text with the experimental setup.

Chapter 2

Related Work

The related work chapter is divided into two sections. First, I review which research has already been done on comics in the area of computer science. Next, the GAN models that are investigated in this thesis are briefly explained.

2.1 Comics Research in Computer Science

2.1.1 Overview

A good place to start is the survey [Augereau et al., 2018], citing about 100 papers and articles with different forms of computer science research on comics. As an experienced comic book artist, I can only confirm what is stated at the beginning of this paper and is repeated in many other papers: "The research about comics is quite challenging because of its nature. Comics contain a mixture of drawings and text. To fully analyze and understand the content of comics, we need to consider natural language processing to understand the story and the dialogues and computer vision to understand the line drawings, characters, locations, actions, etc. A high-level analysis is also necessary to understand events, emotions, storytelling, the relations between the characters, etc."[Augereau et al., 2018]

Comics are a narrative art form consisting of a complex mixture of text and image. It is characterized by a succession of still images separated by a gutter. These images, called panels, are arranged in different sizes on pages to create a functional page layout. The sequence of panels determines the narrative time and the told time. The gutters determine the extent to which the readers supplement the story with their imagination. The drawing style is often a synthesis of reality and in turn, relies on the reader's imagination. Text balloons, frames with a narrator's voice, and sound imitations round out the whole.

The nature of this narrative art makes automatic generation quite challenging. But not only their nature is a hurdle, but also the lack of good and sufficiently large data sets poses difficulties. As stated in [Augereau et al., 2018]: "[...]the high variety of drawings and the scarceness of labeled data make the task harder than natural images.[...] Large datasets with ground truth information such as layout, characters, speech balloon, text, etc. are not available so using deep learning is hardly possible

in such conditions." The paper organizes the research into three categories: content analysis, content generation/adaptation, and user interaction. Only the content generation category is of interest in the context of this thesis.

2.1.2 Panel Composition

In the world outside AI, comic panels are composed by sketching. Based on collected photo material or preparatory drawings, a situation is staged in which characters are placed against a certain background. Objects are added and set elements are placed in front of the characters. Before or at the same time, the camera position and lens angle are determined. This may seem strange for a medium that is not filmed, but the analogy is equally valid. Scenes, for example, are placed in frog perspective or close-up, depending on the narrative or visual added value.

Some papers referred to by [Augereau et al., 2018] suggest a similar composition of panels, albeit in a greatly simplified form. They attempt to generate panels through a collage of characters on backgrounds. [Cao et al., 2014] suggests using a probabilistic graphical model and collecting data from eye-tracking experiments to compose pages and panels to provide an optimal reading experience. While this approach is interesting, its strong limitations become immediately apparent. It works with a limited database of characters, backgrounds, and speech bubbles. In the long run, this results in comics that always look the same, which risks becoming very monotonous. Likewise, [Veale, 2022] proposes a similar technique of panel composition through a combinatorial approach and has the same problem. While the idea behind the synthesis of panels with a collage is interesting, the visual richness of the results is always limited by the size of the underlying database of existing drawings of characters, objects, and backgrounds. These methods don't generate any new drawings.

2.1.3 Panel Generation

An alternative approach is the generation of drawings with GANs (Generative Adversarial Networks). Recent years have seen staggering progress in this domain. Since the launch of the first GAN algorithm by Ian Goodfellow in 2014 [Goodfellow et al., 2014], new architectures, better learning methods, and all kinds of tweaks have been proposed at a rapid pace resulting in a significant improvement of its performance. In the research on the creation of comics with GANs, two strands stand out. The first one does unconditional and/or conditional comic panel generation by modeling an existing dataset. The second one takes a detour via pictures by modifying the style of real images until it becomes more like a comic panel.

[Jin et al., 2017] trained a DRAGAN [Kodali et al., 2017] on 42.000 anime[1] face images. Their results are very good as shown in figure 2.1. Because the dataset is tagged, the model of [Jin et al., 2017] can perform conditional generation. Certain visual face features, like the length and color of the hair, can be determined by changing the parameters of the condition. [Morris, 2021] and [Proven-Bessel et al., 2021]

[1] Anime is a Japanese term for animation.

Figure 2.1: Generated anime faces by [Jin et al., 2017], the picture was taken from the paper.

both performed an experimental analysis of different GANs on single comic panels of the newspaper series Dilbert. These two papers are in line with my thesis and I will refer to them when I evaluate my results.

[Proven-Bessel et al., 2021] also attempts text-to-comic generation with AttnGAN [Xu et al., 2017] as it "seeks to further the area of scientific comic research" but does this only with limited success. I do agree with its analysis: "Features of comics are different from those in (photographic) images. Generally, they are simpler and bolder. Black lines distinguish all characters and objects, characters each have unique hair, facial features, and clothes. Backgrounds are often a single block color, and do not contain the same level of variation and detail." [Morris, 2021] makes an interesting remark on the quality of a dataset: "[...] the results of image synthesis given a specific GAN architecture are highly dependent on the training data. In short, there is no way to predict how these networks will perform on comic data without an analytic comparison of their results." That is what this thesis is focussing on.

[Chen et al., 2020] has a completely different approach to generating comic panels. Starting from photographic images, the panels are created with neural style transfer. Since the training was done on manga[2] and anime, the style is dominated by the look of Asian comics. This approach is compelling because of the existence of very powerful text-to-image generators trained on the vast amount of photos on the internet. One could assemble an autonomous comic machine starting from a text-generating algorithm, generating photographic pictures and then changing their style with this model. There are however some drawbacks that hinder the usability of this technique in practice. First of all, photos are not comic panels and neither are stylized photos. Comic panels are characterized by a synthesis of the image in which backgrounds are purged of unnecessary details so as not to interfere with the reader's attention. Comic panels created with AnimeGAN still contain as many unnecessary details as the original photo. Secondly, as can be seen in figure 2.2, the network seeks to form closed detoured areas filled with a single color. Sometimes this goes wrong, because of misinterpretation of artifacts such as hair hanging in

[2]Manga is a Japanese term for comics.

(a) Original picture (b) Stylized with AnimeGAN

Figure 2.2: AnimeGAN [Chen et al., 2020] creates comic panels in manga style with neural style transfer. Picture ©Marnix Verduyn. Style transformation created with [Long, 2020]

front of someone's face.

2.2 Generative Adversarial Networks

2.2.1 Surveys

Once again, a good place to start a search for GAN architectures for unconditional image generation, are overview papers. Since the introduction of the GAN network [Goodfellow et al., 2014], this field of research has received wide attention. The following surveys provide an overview of what has since come to be called the GAN Zoo. [Pan et al., 2019], [Creswell et al., 2018] and [Liu et al., 2020] are structured identically. A distinction is made between GANs based on an optimized architecture and GANs trying to optimize their objective function. An overview is given of all the tricks to stabilize the training of GANs and all the evaluation metrics. Finally, the different applications in the domain of image synthesis are discussed. [Wiatrak et al., 2019] focuses solely on the stabilization methods and provides a comprehensive overview of which methods can be found in the literature. I will mention some of them because they are part of the networks used in this study.

2.2.2 GAN

Generative Adversarial Networks (GANs) are a type of generative model that became very popular in a short time due to their exceptional ability to model complex real-world data. In this thesis, I limit myself to image synthesis tasks, more specifically

comic book drawings. But GANs can also handle video, natural language, and other kinds of data. As stated in many texts, but adapted to the research question as discussed in chapter 1: "A common analogy, apt for comic drawings, is to think of one network as a comic art imitator and the other as a comic art expert. The imitator, known in the GAN literature as the generator, G, creates imitations, to make new comic panels in the same style as the original master. The expert, known as the discriminator, D, receives both imitations and real (authentic) comic panels and aims to tell them apart. Both are trained simultaneously, and in competition with each other." [Creswell et al., 2018]. The two models are typically neural networks.

As explained in the original paper [Goodfellow et al., 2014]: "D and G play the following two-player minimax game with value function $V(D, G)$:

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

In practice, rather than training G to minimize the last term, it is trained to maximize $\log D(G(z))$ which provides a more sufficient gradient.

Or, as clearly explained in [Wiatrak et al., 2019]: "The generator is given a Gaussian or uniform random noise $z \sim p_z(z)$, and transforms it in such a way that $G(z)$ resembles a target distribution $p_{data}(x)$. Specifically, the generator is trained to maximize the expected log-probability with which the discriminator classifies $G(z)$ as a real sample, $\mathbb{E}_{z \sim p_z(z)}[\log D(G(z))]$. Conversely, the discriminator is provided with a set of unlabeled samples from $G(z)$ and $p_{data}(x)$ and is trained to distinguish between the generated (fake) and real samples. Formally, the discriminator is trained to maximize the expected log-probability $\mathbb{E}_{x \sim p_{data}(x)}[\log D(x)]$ while simultaneously minimizing $\mathbb{E}_{z \sim p_z(z)}[\log(D(G(z)))$."

The training happens in an alternating manner. Goodfellow proposed originally [Goodfellow et al., 2014]: "[...] we alternate between k steps of optimizing D and one step of optimizing G. This results in D being maintained near its optimal solution, so long as G changes slowly enough." Other alternating schedules have been tried. In general D is not trained till optimality at each step. [Wiatrak et al., 2019]: "Such a highly accurate discriminator, [...] squashes the loss function to 0, producing gradients close to zero, which provides little feedback to the generator, slowing or completely stopping the learning."

At convergence, the interesting result is that G becomes a "[...] generative machine, i.e. a model that does not explicitly represent the likelihood, yet can generate samples from the desired distribution."[Goodfellow et al., 2014]

2.2.3 Instability and a Performance Measure

However, GANs are known to be unstable in training. From [Salimans et al., 2016] we learn: "[...] training GANs requires finding a Nash equilibrium of a non-convex game with continuous, high-dimensional parameters. GANs are typically trained using gradient descent techniques that are designed to find a low value of a cost function, rather than to find the Nash equilibrium of a game. When used to seek a Nash equilibrium, these algorithms may fail to converge."

[Wiatrak et al., 2019] resumes it as follows: "Common failures include mode collapse in which the generator maps different inputs to the same class of outputs which results in highly non-diversified samples; non-convergence due to oscillatory and diverging behaviors during the training of both generator and discriminator; and vanishing or exploding gradients leading to slow learning or a complete stop of the learning process [Salimans et al., 2016]." "Furthermore, the evaluation and benchmarking of GANs has been hampered by the lack of a single universal and comprehensive performance measure." "These problems are often highly linked with each other and addressing them often takes the character of a trade-off. A crucial challenge is the accurate estimation of the generalization capabilities of the discriminator, which in turn would make it possible to properly evaluate the generator. A final obstacle is the lack of a comprehensive evaluation metric. At the current state, several evaluation metrics have been proposed, such as Inception or Frechet Inception Distance (FID). Nevertheless, none of them offer a unique solution."

Amongst the techniques for stabilizing the GAN training, we find modified architectures, modified loss functions, modified gradient optimization and several general stabilization heuristics that are deployed in the GANs studied in this thesis: DCGAN, WGAN, and StyleGAN2-ADA. [Wiatrak et al., 2019]: "Currently, the approach showing the highest empirical improvement is the architecture type."

2.2.4 DCGAN

It may seem odd that a model like DCGAN is part of this research. After all, it was proposed in 2015. That is seven years ago and in a rapidly evolving science like AI, that is an eternity. But DCGAN is an interesting model because it is simple, trains relatively stably, and delivers quite good results. Moreover, it trains quickly on a custom dataset with an ordinary computer. You do not have to perform transfer learning on a model that is pre-trained with gigantic datasets on supercomputers. As stated in [Wiatrak et al., 2019]: "One of the most popular and successful architecture-variant GANs are inspired by computer vision techniques, specifically Convolutional Neural Networks (CNNs). An example that has been shown to significantly improve the quality of generated images is the Deep Convolutional GAN (DCGAN)[Radford et al., 2015]. DCGAN applies a series of convolutional operations including a spatial up-sampling operation to improve the generator. Having shown to enhance the quality of the produced samples solely through architectural changes, DCGAN is often treated as a baseline for modeling other GAN variants ."

The generator G used in this thesis starts from a 100-dimensional random $\mathbf{z}$ vector. After 4 fractionally-strided convolution layers, an image of resolution 64^2 is generated (see figure 2.3). The discriminator D has a similar but reversed architecture, ending with a single probability value.

[Wiatrak et al., 2019] : "In their original formulation, [Goodfellow et al., 2014] have shown that GANs optimize the Jensen-Shannon divergence (JSD)

$$JSD(Pr||Pg) = \frac{1}{2}KL(\mathbb{P}_r||\mathbb{P}_A) + \frac{1}{2}KL(\mathbb{P}_g||\mathbb{P}_A) \tag{2.2}$$

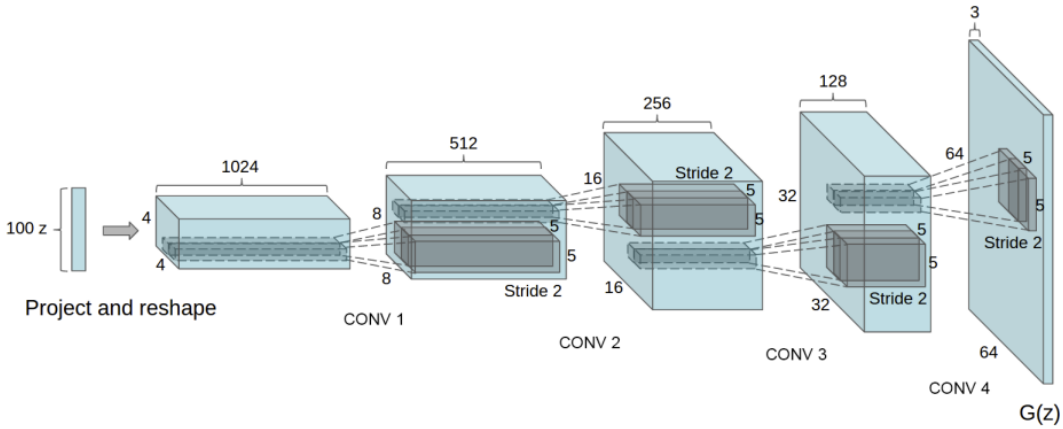


Figure 2.3: DCGAN Generator (picture taken from the paper [Radford et al., 2015]).

where KL denotes the Kullback-Leibler divergence and $\mathbb{P}_A$ is defined as the "average" distribution $\mathbb{P}_A = \frac{P_r+P_g}{2}$." The trouble is that during training of the DCGAN, we only have an idea of the losses of G and D. In an optimal situation they will evolve towards $0, 5$ and we hope that at the same time the discrepancy between the model distribution and the real distribution will be minimized. But the model itself is not driven by the minimization of a metric quantifying this discrepancy. This is the main objective of WGAN.

2.2.5 WGAN

Since the real data distribution and the model distribution are both supported by low dimensional manifolds in a higher dimensional space, "It is then unlikely that the model manifold and the true distribution's support have a non-negligible intersection, and this means that the KL distance is not defined (or simply infinite)" [Arjovsky et al., 2017]. In this paper, the proof is shown that the Wasserstein distance has nicer properties than the JS or KL distance. "The Wasserstein distance or Earth-Mover (EM) distance is defined as:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||], \tag{2.3}$$

where $\prod(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively $\mathbb{P}_r$ and $\mathbb{P}_g$. Intuitively, $\gamma(x, y)$ indicates how much "mass" must be transported from x to y in order to transform the distributions $\mathbb{P}_r$ into the distribution $\mathbb{P}_g$. The EM distance then is the "cost" of the optimal transport plan. [...] However, the infimum in 2.3 is highly intractable." and through the Kantorovich-Rubinstein duality which uses a K-Lipschitz function, the paper comes up with a simple but brute way to approximate this Wasserstein distance, by model weight clipping. In this algorithm, the D is trained till optimality at each step. Again from [Arjovsky et al., 2017]: "The argument is simple, the more we train the critic,

the more reliable gradient of the Wasserstein we get [...]" and "Perhaps, more importantly, the fact that we can train the critic till optimality makes it impossible to collapse modes when we do." Because the loss function is an estimate of the EM distance, contrary to DCGAN we now have a meaningful loss metric as an indication of the quality of the generated samples during training. The loss should decrease consistently during training while the sample quality should increase.

To end this section on WGAN, it is worth mentioning a remark noted in [Wiatrak et al., 2019]: "[...] loss function variant GANs, although often well justified, such as WGAN, show that their results could also be attained by appropriate hyperparameter optimization of architecture variant GANs such as DCGAN. A comprehensive comparison of GANs has demonstrated a small improvement in general metrics by state-of-the-art loss type GANs in comparison to architectures such as DCGAN, and that it can be more worthwhile to focus on tuning hyperparameters rather than introducing novel methods."

2.2.6 StyleGAN

When StyleGAN [Karras et al., 2018] was proposed, it represented a major leap forward in image quality. The authors of the paper argued that "the generators continue to operate as black boxes, and despite recent efforts, the understanding of various aspects of the image synthesis process, e.g., the origin of stochastic features, is still lacking." They proposed a complete redesign of the model. Two major changes were introduced: the disentanglement of the latent space and the direct control of image features at different scales.

StyleGAN steps away from the idea of a random vector $\mathbf{z}$ which generates an image after it has been propagated through a feedforward network. This time, there are two networks, a mapping network f and a synthesis network g (see figure 2.4). At the input of the synthesis network g, there is a constant tensor. It propagates through the network and creates the image. To make changes to the generated images, the image features are tweaked at different levels in the network. This is done through the mapping network f. And that is where the random vector $\mathbf{z}$ comes back into the picture. Vector $\mathbf{z}$ is taken randomly from the latent space $\mathcal{Z}$ and is now first propagated through a non-linear mapping network f to create a new vector $\mathbf{w}$ in the latent space $\mathcal{W}$. After that, $\mathbf{w}$ is used to control the adaptive instance normalization at each convolution layer.

As explained in more detail in [Karras et al., 2018]: "For simplicity, we set the dimensionality of both spaces to 512, and the mapping f is implemented using an 8-layer MLP. [...] Learned affine transformations then specialize $\mathbf{w}$ to *styles* $\mathbf{y} = (\mathbf{y}_s, \mathbf{y}_b)$ that control adaptive instance normalization (AdaIN) operations after each convolution layer of the synthesis network g. The AdaIN operation is defined as

$$AdaIN(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i}\frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i} \tag{2.4}$$

where each feature map $\mathbf{x}_i$ is normalized separately, and then scaled and biased using the corresponding scalar components from style $\mathbf{y}$. Thus the dimensionality of

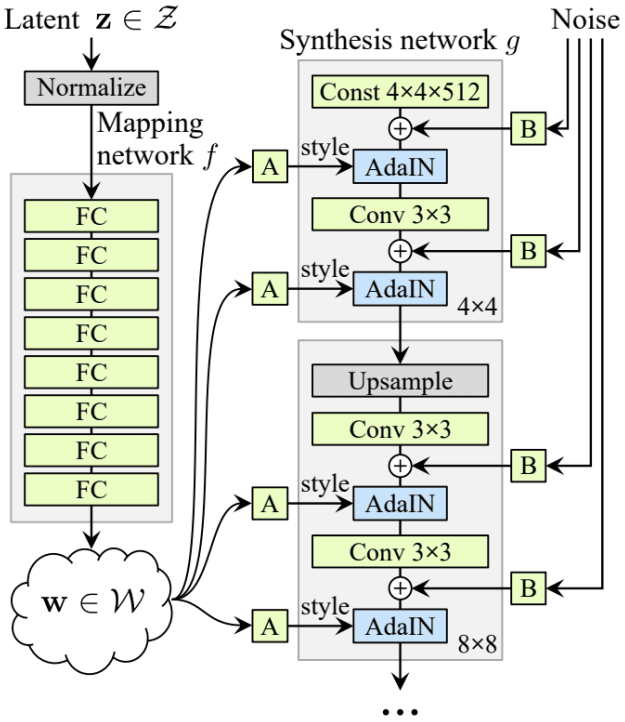


Figure 2.4: StyleGAN: a style-based generator with two networks f and g (picture taken from the paper [Karras et al., 2018]).

$\mathbf{y}$ is twice the number of feature maps on that layer."

The reason for using a mapping network is explained in the paper with an example from the faces dataset. Nowadays men with long hair are rather rare. This is a clear example of the entangling of two style features (gender and the length of hair). In the latent space $\mathcal{Z}$ of a model trained on a faces dataset, there are regions for which there are no or very few examples of long-haired men. But the random vector $\mathbf{z}$ is sampled from a normal distribution. So the classical network G will have to learn a deformation of $\mathcal{Z}$ in its mapping to avoid these no-go zones. In StyleGAN, this is decoupled thanks to the mapping network f. The latent space $\mathcal{Z}$ is first transformed into the latent space $\mathcal{W}$ of the same dimension 512, so that much of this warping can be avoided afterward in the generator.

Additional noise is injected at each level that has been previously multiplied by a learned scaling factor. The explanation for this is as follows [Karras et al., 2018]: "We hypothesize that at any point in the generator, there is pressure to introduce new content as soon as possible, and the easiest way for our network to create stochastic variation is to rely on the noise provided. A fresh set of noise is available for every layer, and thus there is no incentive to generate the stochastic effects from earlier activations, leading to a localized effect."

The loss functions are WGAN-GP and a non-saturating loss with R_1 regularization.

The authors also avoid sampling from regions of $\mathcal{W}$ with lower probability with the so-called truncation trick, but only for the lower resolutions of the generated image. This sampling from a truncated latent space improves image quality but also limits the amount of variation.

No other stabilization methods are used. As stated in [Karras et al., 2018]: "We do not modify the discriminator or the loss function in any way, and our work is thus orthogonal to the ongoing discussion about GAN loss functions, regularization, and hyperparameters."

2.2.7 StyleGAN2

StyleGAN2 [Karras et al., 2019] was introduced as a solution to some artifacts in the images generated by StyleGAN. The results are indeed significantly better. The main changes are a revision of the instance normalization, a redesign of the normalization used in the generator, and the abandonment of the progressive gan structure. The architecture of StyleGAN is altered as follows. First, modulation and normalization are removed from the flow of instances to directly affect the weights in the convolution. As explained in the paper: "The main idea is to base normalization on the expected statistics of the incoming feature maps but without explicit forcing. [...]The modulation scales each input feature map of the convolution based on the incoming style, which can alternatively be implemented by scaling the convolution weights:

$$w'_{ijk} = s_i \cdot w_{ijk}, \tag{2.5}$$

where w and w' are the original and modulated weights, respectively, s_i is the scale corresponding to the ith input feature map, and j and k enumerate the output feature maps and spatial footprint of the convolution, respectively. Now, the purpose of instance normalization is to essentially remove the effect of s from the statistics of the convolution's output feature maps. We observe that this goal can be achieved more directly."

The normalization comes down to:

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

where ϵ is a small constant to avoid numerical issues. Figure 2.5 shows the revised architecture.

A second important change comes from the realization that perceptual path length (PPL) is a metric linked to image quality. As stated in [Karras et al., 2019]: "We observe a correlation between perceived image quality and perceptual path length (PPL), a metric that was originally introduced for quantifying the smoothness of the mapping from a latent space to the output image by measuring average LPIPS distances between generated images under small perturbations in latent space." LPIPS stands for Learned Perceptual Image Patch Similarity and is "calculated as a weighted difference between two VGG16 embeddings."[Karras et al., 2018]. Although it is not immediately obvious to the authors why PPL should correlate with image quality,

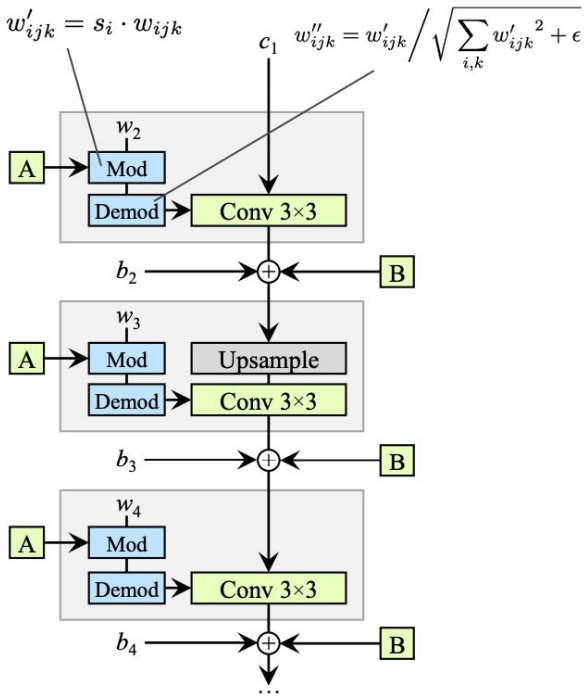


Figure 2.5: StyleGAN2: a "demodulation" operation is applied to the weights associated with each convolution layer (picture taken from the paper [Karras et al., 2019]).

they build a regularizer in the generator to encourage it to obtain a well-conditioned mapping from the latent space $\mathcal{W}$ to the image space. [Karras et al., 2019]: "In practice, we notice that path length regularization leads to more reliable and consistently behaving models, making architecture exploration easier." However, this introduces a trade-off between PPL and FID in datasets that are not well structured. This path length regularization can also not be tuned in the PyTorch StyleGAN2 implementation I used.

StyleGAN was based on the ideas of the progressive growing GAN [Karras et al., 2017] but StyleGAN2 steps away from this architecture. Experiments showed that G benefited from skipped connections and D from residual nets. The paper proves that the philosophy behind progressive growth was nevertheless not lost. Instead of imposing it manually, the generator seems to focus itself initially on low resolutions to pay more attention to the high resolutions afterward.

As a result, it also showed easier to perform projections of images in the latent space. This is a technique in which the optimal latent vector $\mathbf{w}$ is searched for a provided image. If the image is previously generated, this succeeds almost perfectly. However, there are clear deviations when it is a real image that was not in the

Figure 2.6: A real image (upper-left) and a projected image (upper-right) with an FFHQ pre-trained StyleGAN2. A style-transferred image (lower-left) between the real image and the comic character (lower-right). Real image: ©Cyrus Pâques

dataset. This is illustrated in Figure 2.6 where we see a projection for an image with StyleGAN2 FFHQ that was not in the dataset. Its basic characteristics are identical but in the details, there are obvious differences.

Style transfer is also one of the possibilities. This is done by combining the $\mathbf{w}$ vectors of two projected images into one new vector $\mathbf{w}$. The embeddings happen in the extended version in the latent space $\mathcal{W}^+$. "$\mathcal{W}^+$ is a concatenation of 18 different 512-dimensional $\mathbf{w}$ vectors, one for each layer of the StyleGAN architecture that can receive input via AdaIn"[Abdal et al., 2019b]. The lower half of the first vector then affects the coarse features of the image and the upper half of the second vector determines the fine features. This is illustrated in figure 2.6 where we see a style transfer between a cartoon character (lower right) and the original image (upper left). The coarse image features such as the round head, smile with bare teeth, big eyes, and long hair are taken from the comic book character. Fine image features such as stubble, hair color, and skin color are taken from the original image.

2.2.8 StyleGAN2-ADA

As mentioned earlier, training is often hindered by discriminator overfitting. As D becomes overconfident, it no longer yields interesting gradients from which G learns something useful. [Karras et al., 2020] is an interesting study of the effects of a wide range of augmentations to the images, resulting in a better converging model, without leaking the augmentations into the generated images. In a very straightforward approach, augmentation is applied to both the fake and real images before being judged by the generator. Explained with a clear metaphor in [Karras et al., 2020]: "Discriminator augmentation corresponds to putting distorting, perhaps even destructive goggles on the discriminator, and asking the generator to produce samples that cannot be distinguished from the training set when viewed through the goggles." The only condition is that the transformations are invertible. This does not mean that the individual image manipulations have to be undoable. It means that it still has to be possible to reason about the original distribution. As illustrated in [Karras et al., 2020]: "[...] random rotations chosen uniformly from $\{0°, 90°, 180°, 270°\}$ are not invertible: it is impossible to discern differences among the orientations after the augmentation. The situation changes if this rotation is only executed at a probability $p < 1$: this increases the relative occurrence of $0°$, and now the augmented distributions can match only if the generated images have the correct orientation." Experiments have led to a maximum value for this probability $p = 0.8$ under which it is unlikely for leaks to happen. An augmentation pipeline is proposed, which is composed of pixel blitting[3], geometric and color transforms. To some extent, it remains searching because [Karras et al., 2020]: "[...] the optimal augmentation strength depends heavily on the amount of training data, and not all augmentation categories are equally useful in practice." ADA, which stands for Adaptive Discriminator Augmentation, has even a built-in tuning of augmentation strength so that overfitting is optimally avoided.

For comic art, there should be no leaking of the augmentations at all. It is not a real problem in the case of a horizontal flip or an isotropic scaling. Those are transformations that are frequently used during the creation process. But color corrections or $90°$ turns are out of the question. In comics, there is often a fixed color palette and you don't want the characters to appear upside down in the panels unless it is intentional.

2.3 Conclusion

The ultimate moonshot is of course a fully autonomous comic generator. But the complexity of the 9th art[4] does not permit such a challenge for now. As stated in [Proven-Bessel et al., 2021]: "Up until recently no existing machine learning algorithms have been developed to create comic illustrations based on descriptions of illustrations, or the dialogue in comics." In my study of the literature, I did not

[3] Pixel blitting is a term for transformations like x-flip, $90°$ rotations, and integer translations.

[4] Comics are called "The 9th Art" (after Architecture, Sculpture, Painting, Music, Dance, Poetry, Film and Television)

encounter research that came to satisfactory results for this objective. Hence, this thesis is also limited in its scope to tackle the long-term objective step by step. I perform experimental research on the unconditional generation of comic art as a starting point. The first thing needed for a machine to make comics is to be able to draw panels in a specific style. Starting with DCGAN as a good entry point, WGAN is briefly inspected. Then StyleGAN2 is examined more thoroughly. The latter model has recently received one more update (StyleGAN3) but still produces state-of-the-art results.

Chapter 3

Comic Art Generation using DCGAN

This chapter describes the experiments of unconditional image creation with DCGAN on the dataset of Kinky & Cosy.

3.1 Dataset

DCGAN as presented in [Radford et al., 2015] generates images of size 64×64. The paper describes training on three datasets, LSUN, ImageNet-1k and a faces dataset. The latter is also the smallest of the three and contains 350.000 images. My dataset contains 56.234 images. These are stills from the Kinky & Cosy animation series that have been cropped left and right to be square and then resized to 64×64. As noticeable in figure 3.1, the diversity of this data set is quite large. Certainly larger than a set of centered and frontal faces where the eyes, nose, and mouth are always roughly in the same place. In Kinky & Cosy you have all kinds of diverse situations. Characters vary, camera angles change, and backgrounds differ from image to image. Because of the cropping of the stills, some characters are half-cut off. The resolution is low so many details are lost and the typical hard black lines around the figures are blurred. This is not an ideal situation for the generation of usable images but initially, this is just a test to evaluate the performance of the model.

3.2 Architecture

For the implementation of DCGAN, I used PyTorch [Inkawhich, 2022]. The structure and parameters are all taken from the initial paper on DCGAN [Radford et al., 2015]. The generator starts from a 100-dimensional vector $\mathbf{z}$ and consists of four fractionally-strided convolution layers. In between each, a batch normalization occurs. The activation functions are $ReLu()$, except in the last layer where it is $Tanh()$ to return it to the input data range of [-1,1]. The generator consists of four convolution layers. The activation functions are $LeakyRelu()$. The last layer ends with a $Sigmoid()$ to output the final probability. The batch normalization and the leaky relu functions

Figure 3.1: Training data: stills from the animation movies, cropped left and right.

are there to obtain a healthy gradient flow which is critical for the learning process of both the generator and the discriminator. All model weights are randomly initialized from a normal distribution with mean= 0 and standard deviation= 0.02. There are two separate optimizers, one for the discriminator and one for the generator. Both are Adam optimizers with learning rate= 0.0002 and $\beta_1 = 0.5$.

3.3 Training

The training with batch size 128 takes about 30 minutes to converge on a one GPU NVIDIA RTX A4000. That is after 30 epochs and over 65.000 iterations.

3.3.1 Discriminator

As explained in [Inkawhich, 2022]: "[...] the goal of training the discriminator is to maximize the probability of correctly classifying a given input as real or fake. In terms of Goodfellow, we wish to "update the discriminator by ascending its stochastic gradient"[Goodfellow et al., 2014]. Practically, we want to maximize

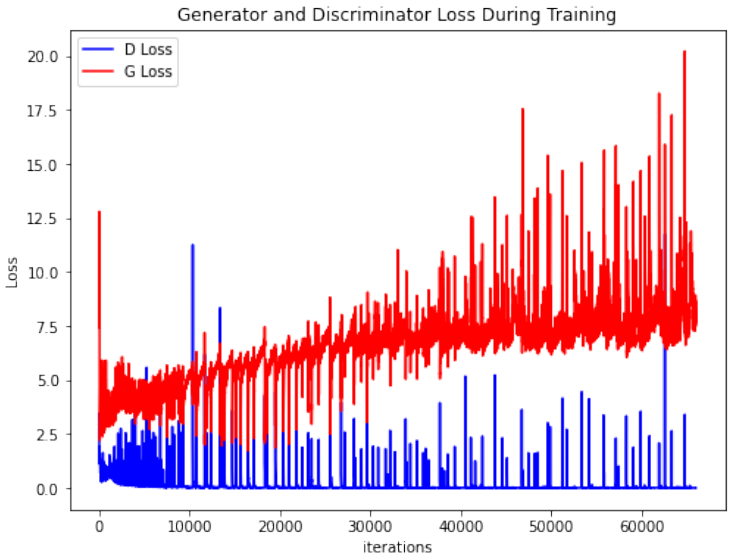


Figure 3.2: Loss versus training iterations for the DCGAN.

$\log(D(x)) - \log(1 - D(G(z)))$. Due to the separate mini-batch suggestion from [Salimans et al., 2016], we will calculate this in two steps. First, we will construct a batch of real samples from the training set, forward pass through D, calculate the loss ($\log(D(x))$), then calculate the gradients in a backward pass. Secondly, we will construct a batch of fake samples with the current generator, forward pass this batch through D, calculate the loss $\log(1 - D(G(z)))$, and accumulate the gradients with a backward pass. Now, with the gradients accumulated from both the all-real and all-fake batches, we call a step of the discriminator's optimizer."

3.3.2 Generator

As explained in [Inkawhich, 2022]: "[...] we want to train the Generator by minimizing $\log(1 - D(G(z)))$ in an effort to generate better fakes. [...] this was shown by Goodfellow to not provide sufficient gradients, especially early in the learning process. As a fix, we instead wish to maximize $\log(D(G(z)))$."

3.3.3 Loss during training

Figure 3.2 shows the loss versus the training iterations for the DCGAN. The graph shows inverted values since all log values are negative. As expected, the generator loss increases at first but then decreases and converges to a value. The discriminator loss however evolves to a very small value. What we see here is a discriminator that becomes too clever, leaving the generator with nothing to learn from the discriminator. After about 30.000 iterations, the learning process also begins to oscillate tremendously. Successive versions of the generator can vary a lot in performance.

Figure 3.3: Generated images with DCGAN trained on cropped stills (size 64×64 pixels).

3.4 Evaluation

Figure 3.3 shows the resulting fake images, which is the output of the generator on a batch of fixed noise vector $\mathbf{z}$. We can see that DCGAN performed approximate modeling of the data distribution compared to the original images. The model generates images with the right colors. But based on a purely visual evaluation, it is clear that these images are not useful in practice. Characters can be distinguished but can hardly be identified. There is no sharp border with a black line and the colors are not neatly flat, there are too many color gradients.

The generated sample on the 4th row and 5th column in Figure 3.3 is an exception. There, a clear drawing of Kinky & Cosy on a treadmill in the gym seems to form. Not coincidentally, this drawing occurs exceptionally often in the training dataset. It is a series of stills of an episode where the two main characters run an entire episode on a treadmill. In the data distribution, this represents a large probability peak at the position of this image vector. As discussed in Appendix A, this is one of a few anomalies in the dataset.

Figure 3.4: New training data: selected stills from the animation movies, focusing on the main characters Kinky & Cosy.

3.5 Improvement by Adjusting the Dataset

The images generated in [Radford et al., 2015] after training on a faces dataset are better in quality. This is closely linked to the quality of the dataset. The cropped frames from the Kinky & Cosy animation movies are far too diverse for their limited number. In other words, there are too many 'holes' in the data distribution. By 'holes' I mean zones where no information is present at all so that when the data distribution is modeled, non-sensical images are created.

One possible solution is to adjust the dataset to have less variance. In a second approach, I scanned the original animation frames looking for the main characters Kinky & Cosy, and detoured the images so that only the twins are in focus. A sample of the new dataset is shown in figure 3.4. It now consists of 8.543 samples. This is much less than the previous dataset, but the hope is that due to the decreased variance, the results will be more visually interesting.

Although the loss functions show the same trend during training, the results are indeed much better as shown in Figure 3.5. The colors have lost some saturation, but the generated images now contain distinguishable and almost identifiable characters

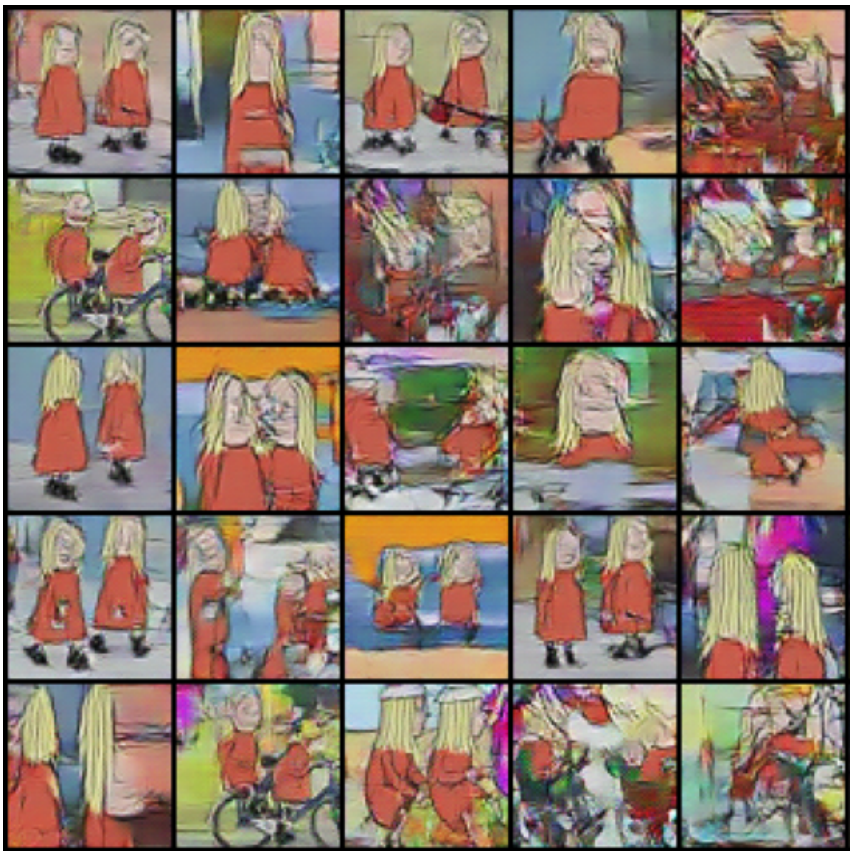


Figure 3.5: Generated images with DCGAN trained on images focussed on Kinky & Cosy (size 64×64 pixels).

that resemble Kinky & Cosy. The black lines are still not sharply delineated and the color planes are not even throughout but there is a noticeable improvement. However, if we carefully compare the real dataset with the fake images, we notice that close-ups no longer occur. Also, the out-of-the-ordinary images that occasionally appear amongst the real images are not generated. Examples are Kinky & Cosy with a bookbag on their back or wearing balaclavas (see figure 3.4). So it seems that the model data distribution is less diverse than the real data distribution.

To get a better idea of the modeling of the data distribution, we can look at some interpolations between random points in the latent space (see figure 3.6). This seems to confirm the suspicion that there are 'holes' in the data distribution of the training dataset. Take for example the interpolation on the fourth row. The image on the left is Kinky & Cosy sitting on the cart of a roller coaster and the image on the right is Kinky & Cosy on a bicycle. However, in the transition from the first image to the second, we see some non-sensical images. The difference is large if we compare this to the results of a DCGAN trained on faces. When performing an interpolation, we see different kinds of new faces between a first face and a second face. This is an indication that the current dataset still does not seem to be sufficiently condensed.

Figure 3.6: Linear interpolation between latent points for DCGAN.

Another reason could be that linear interpolation is not the most appropriate way of interpolating in a normalized input space. As stated in [White, 2016]: "Frequently linear interpolation is used, which is easily understood and implemented. But this is often inappropriate as the latent spaces of most generative models are high dimensional (> 50 dimensions) with a Gaussian or uniform prior. In such a space, linear interpolation traverses locations that are extremely unlikely given the prior." This is based on the failing intuition in high dimensions. [Domingos, 2012] illustrates this with a juicy metaphor: "After overfitting, the biggest problem in machine learning is the curse of dimensionality. [...] Our intuitions, which come from a three-dimensional world, often do not apply in high-dimensional ones. In high dimensions, most of the mass of a multivariate Gaussian distribution is not near the mean, but in an increasingly distant "shell" around it; and most of the volume of a high-dimensional orange is in the skin, not the pulp." A better technique would be a spherical interpolation. Quaternion spherical linear interpolation (SLERP) is an extension of linear interpolation along a plane to spherical interpolation in three dimensions. However, I tried this approach but that did not improve the results.

3.6 Higher resolution

After training on images of 64^2 pixels, I modified the model with an additional layer in G and D to train on the same dataset but at a resolution of 128^2. This did not produce satisfactory results. The model became unstable. Despite several attempts to train with different values for the learning rate, I did not obtain results that

matched the level of the 64^2 images. The problem was situated with an over-confident discriminator from which the generator could soon learn nothing. The literature then refers to the techniques that stabilize the learning process, as discussed in Chapter 2. The next chapter looks at one of these techniques, a modified loss function.

3.7 Conclusion

DCGAN is a simple architecture with powerful results for large datasets with limited variance. It is an interesting entry-level model to get acquainted with how GANs work. However, applied to the Kinky & Cosy dataset, it does not provide useful results. A more recent model should be investigated.

Chapter 4

Comic Art Generation using WGAN

This chapter describes the experiments of unconditional image creation with WGAN on the dataset of Kinky & Cosy.

4.1 Dataset

The WGAN I tested generates images of size 32×32. The experiment is performed on the first dataset with maximum diversity (figure 3.1).

4.2 Architecture

For the implementation of WGAN, I used PyTorch [Saswat, 2020]. The structure and parameters are all taken from the initial paper on WGAN [Arjovsky et al., 2017]. The generator starts from a 100-dimensional vector $\mathbf{z}$ and consists of four fractionally-strided convolution layers. In between each, a batch normalization occurs. The activation functions are $ReLu()$, except in the last layer where it is $Tanh()$ to return it to the input data range of [-1,1]. The generator consists of four convolution layers. The activation functions are $LeakyRelu()$. The last layer ends with a $Sigmoid()$ to output the final probability. The batch normalization and the leaky relu functions are there to obtain a healthy gradient flow which is critical for the learning process of both the generator and the discriminator. All model weights are randomly initialized from a normal distribution with mean$= 0$ and standard deviation$= 0,02$. There are two separate optimizers, one for the discriminator and one for the generator. Both are RMSprop optimizers with learning rate$= 0,00005$. The model performs batch normalization at each layer. The weight clip value $= 0,02$

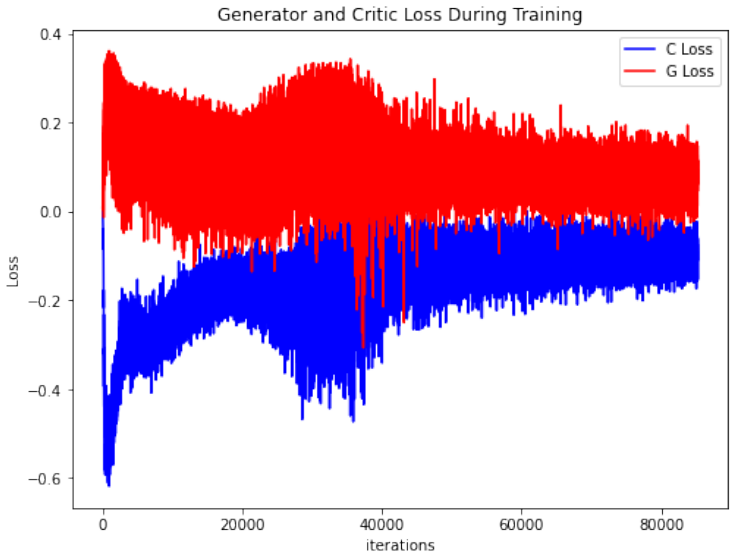


Figure 4.1: Loss versus training iterations for the WGAN.

4.3 Training

The training with batch sizes 64 takes about 30 minutes to converge on a one GPU NVIDIA RTX A4000. That is after 60 epochs and over 90.000 iterations.

4.3.1 Discriminator becomes Critic

Contrary to DCGAN, here the discriminator is trained till optimality, as explained in [Arjovsky et al., 2017]: "The fact that the EM distance is continuous and differentiable a.e. means that we can (and should) train the critic till optimality. The argument is simple, the more we train the critic, the more reliable gradient of the Wasserstein we get, which is actually useful by the fact that Wasserstein is differentiable almost everywhere." The authors even choose for a different name, D is no longer called a discriminator but a critic. As they explain: "The discriminator learns very quickly to distinguish between fake and real, and as expected provides no reliable gradient information. The critic, however, can't saturate and converges to a linear function that gives remarkably clean gradients everywhere.[...] Perhaps, more importantly, the fact that we can train the critic till optimality makes it impossible to collapse modes when we do." [Arjovsky et al., 2017]. The number of iterations to update this critic is taken $n_{critic} = 5$.

4.3.2 Loss during training

Figure 4.1 shows the loss versus the training iterations for the WGAN. The graph shows inverted values since all log values are negative. The course of the loss is different from what DCGAN showed. The evolution of the critic is more interesting

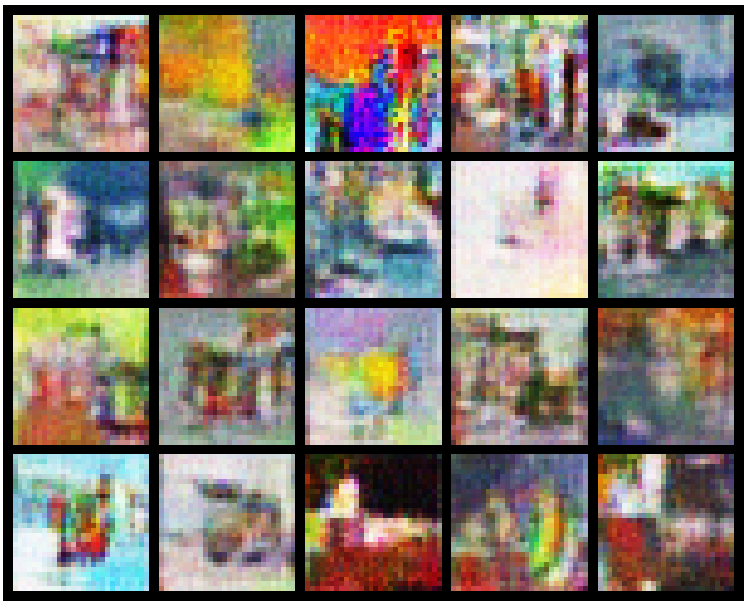


Figure 4.2: Generated images with WGAN trained on cropped stills (size 32×32 pixels).

because it does not saturate. On the other hand, the loss of the generator increases and is also very oscillatory, not indicating good training.

4.3.3 Evaluation

Figure 4.2 shows the resulting fake images, which is the output of the generator on a batch of fixed noise vector $\mathbf{z}$. However, the results show no improvement over DCGAN. The color palette seems to be correctly captured but the shapes are amorphous and do not show any recognizable characters or objects. It could be due to the resolution which is on the very low side. But perhaps it is also due to the explanation of [Wiatrak et al., 2019]: "Wasserstein distance has been shown to significantly improve training stability and convergence, dealing particularly well with the distributions support lying on low dimensional manifolds. However, due to the intractability of the infimum term, the critic f needs to be parameterized and weight clipping needs to be applied. A significant drawback is that weight clipping incurs pathological behavior that contributes to slow convergence and not entirely stable training. These problems have been partly addressed by including a gradient penalty instead of weight clipping in the loss function of WGAN-GP."

4.4 Conclusion

The modified loss function in the model WGAN is an interesting modification to the original GAN model. However, the initial results after training on the Kinky & Cosy dataset are so disappointing that I abandon this path. WGAN-GP could be an

option, but that is also a relatively old model (2017). Hoping to generate visually useful images, I turn my attention in the next chapter to a much more recent model, StyleGAN2-ADA (2020).

Chapter 5

Comic Art Generation using StyleGAN2-ADA

This chapter describes the experiments of unconditional image creation with StyleGAN2-ADA on the dataset of Kinky & Cosy.

5.1 Dataset

StyleGAN can generate images up to a resolution of 1024^2. But to keep the training time within limits, I prepared datasets at resolution 128^2. It is interesting to compare my dataset with the datasets used for training in the StyleGAN papers [Karras et al., 2018] [Karras et al., 2019]. The dataset Flickr-Faces-HQ (FFHQ) has 70.000 images of resolution 1024^2. The datasets in this thesis count 56.234 and 8.547 images (the cropped stills and the stills focused on Kinky & Cosy). But again, the same observations as in Chapter 3 apply here. There is a much larger variance compared to a faces dataset. And there are some images with a large number of almost identical occurrences, as in walk cycles or bicycle cycles. It is expected that this will be noticeable in a lower quality of the generated images.

5.2 Architecture

For the implementation, I used the code shared by [Karras and Hellsten, 2021]. It is the official PyTorch implementation of [Karras et al., 2020].

The loss function for D is equivalent to adversarial loss in original GAN paper [Goodfellow et al., 2014], with additional R_1 regularization [Mescheder et al., 2018]. I started with a γ value of 50 because that's the suggestion of [Karras and Hellsten, 2021]. However, this parameter should be fine-tuned by trial and error. A larger γ value makes the model more stable, which is helpful in case of mode collapse. But then the results are less diverse, so the generated images will not cover the diversity of the real dataset. Since my datasets are characterized by a large diversity, it will be noticeable. A small γ value yields more diversity but can make the model unstable

so ideally training is repeated multiple times, with increased or decreased values of γ to find the optimal results.

The loss function for G is the non-saturating logistic loss $f(x) = \log(sigmoid(x))$ with path length regularization.

Since the loss functions have not changed substantially compared to the DCGAN, the evolution of these parameters during the training does not tell us anything about the discrepancy between the real and fake distribution either. However, the PyTorch implementation makes it possible to have an idea of the degree of convergence, thanks to the FID score (Fréchet Inception Distance) introduced by [Heusel et al., 2017]. FID is a method to measure the quality of generated image samples. It does this by comparing the statistics of generated samples to real samples at the level of the image features. Therefore it uses the output of the last pooling layer of a pre-trained Inception v3 model for 50K generated images and all the images in de training dataset. Calculating this metric does take a considerable amount of time, but it usually confirms what you observe in the successive results, namely that the model no longer evolves after sufficient iterations. That is when convergence is achieved.

The optimizer is an Adam optimizer with $\beta_1 = 0, \beta_2 = 0.99$ and $\epsilon = 10^{-8}$.

5.3 Training

StyleGAN2 is such a complex network that training from scratch takes an incredibly long time. The paper reports [Karras et al., 2019]: "Its total training time was 9 days for FFHQ and 13 days for LSUN CAR." Which is on NVIDIA DGX-1 with 8 Tesla V100 GPUs. For this thesis, I use one GPU NVIDIA RTX A4000. Therefore, it is appropriate to start from a pre-trained model. Based on initial testing (see 5.10) I settled for a model pre-trained on images of My Little Pony (See figure A.3) because the drawing style was the closest to my dataset. Training till full convergence took about a week in a resolution of 128^2. The dataset of the cropped stills is similar to the one shown in figure 3.1 but in a higher resolution (128^2). The second dataset where the stills are cropped around the characters Kinky & Cosy is similar to figure 3.4. The evolution of the D-loss, G-loss, D scores for real images and FID is shown for both datasets in figure 5.1 and 5.2. Both experiments result in a decreasing value for FID. Although the general dataset reaches FID convergence faster (after 1.500 kimg) than the dataset focused on Kinky & Cosy (after 4.000 kimg). The first dataset converges at FID = 46 and the second at FID = 25. This difference is undoubtedly a result of the greater intrinsic diversity of the former dataset relative to the latter. It can also be noted that the D score for images from the training set shows an increasing evolution. Despite the influence of the adaptive discriminator augmentation, a slight form of overfitting still cannot be avoided.

Presumably, a dataset with even less diversity could achieve a lower FID score. Exactly how long the training takes is subject to further research. As the website [Karras and Hellsten, 2021] mentions: "In typical cases, 25.000 kimg or more is needed to reach convergence, but the results are already quite reasonable around 5.000 kimg. 1.000 kimg is often enough for transfer learning, which tends to converge

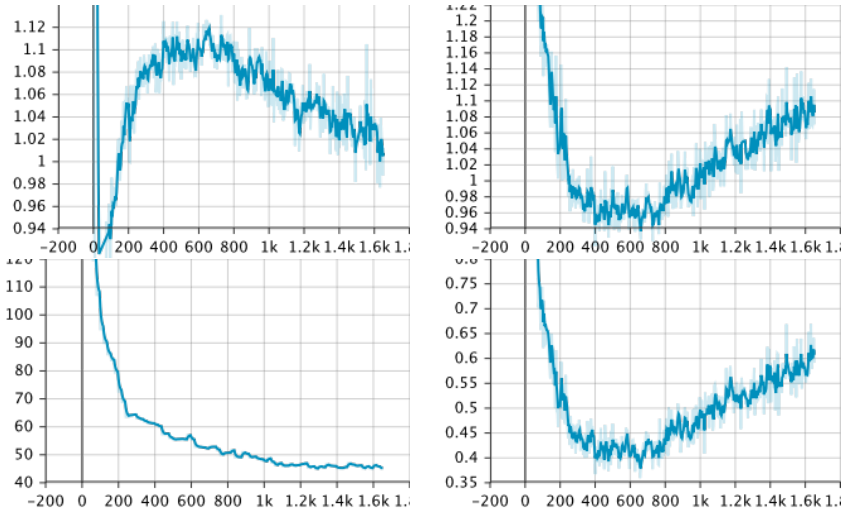


Figure 5.1: Clockwise : D-loss, G-loss, D scores for real images and FID for the dataset of general images.

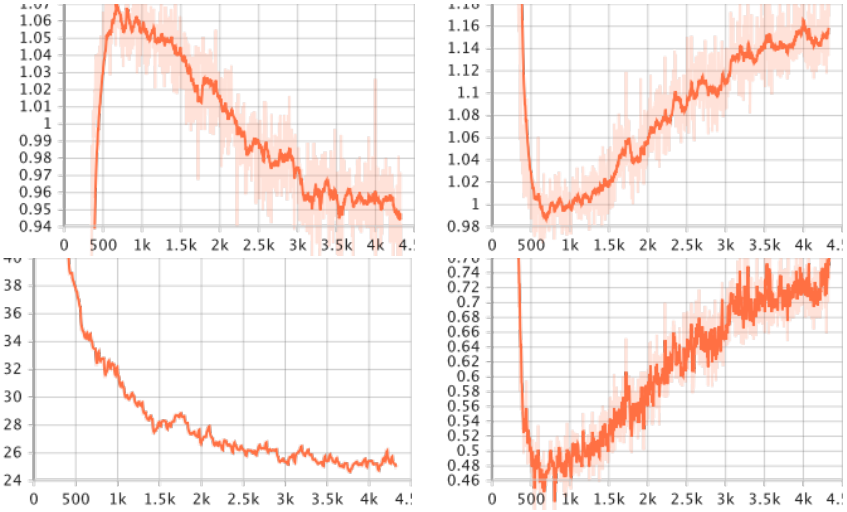


Figure 5.2: Clockwise:D-loss, G-loss, D scores for real images and FID for the dataset of images focussed on Kinky & Cosy.

significantly faster." A less diverse dataset would be desirable to visually improve the results.

Figure 5.3: Generated images with StyleGAN2-ADA trained on general cropped stills (size 128×128 pixels)

5.4 Evaluation

Figure 5.3 shows the results of the first dataset. The most striking impression is the immense improvement over the previously tested models (DCGAN and WGAN). StyleGAN2 has clearly 'understood' the stylistic features of the Kinky & Cosy comics. There is a crisp black line around characters despite the limited resolution and the colors are generally flat on delineated surfaces as they should be, except for some artifacts as short horizontal black stripes. In addition, the colors are correct and not too bland as was the case previously.

A major downside, however, is that none of the drawings are correct. The characters are distorted as are the backgrounds. Limbs are missing, heads are de-duplicated, characters sometimes have more than two eyes, and most strikingly, Kinky and Cosy are apparently triplets instead of twins (see the image on the fourth row and the third column).

Figure 5.4: Generated images with StyleGAN2-ADA trained on images focussed on Kinky & Cosy (size 128 × 128 pixels)

Figure 5.4 shows synthesized images after training on the second dataset. These results are even better which creates hope that it should be possible to achieve the goal of this research: creating usable drawings based on existing comics. The previous dataset produced images where the characters were distinguishable. In this case, they are not only distinguishable, they are even identifiable. In some cases, they are flawless. Having worked with low-cost animation studios in India and China, I received work done by humans that didn't even reach this level of quality. The colors are correct and nicely flat within delineated areas, and the black outline is sharp. The model captured the concept that the two girls are identical except for the glasses.

Compared to the results of [Morris, 2021] and [Proven-Bessel et al., 2021] who trained on a dataset of Dilbert newspaper comics, the Kinky & Cosy generated panels are of a higher visual quality despite the inherent larger variance of the dataset. In contrast, Dilbert is characterized by a high amount of visual repetition (characters

either standing up or sitting at a desk in front of flat colored backgrounds). This FID score is also better: FID = 25 as compared to 100 [Morris, 2021] and 89.2 [Proven-Bessel et al., 2021]. Although comparing FID scores only gives a hint about the difference in performance and not an absolute value judgment, as stated in [Lucic et al., 2017]: "Our main insight is that to compare models it is meaningless to report the minimum FID achieved. Instead, we propose to compare distributions of the minimum achievable FID for a fixed computational budget. Indeed, empirical evidence presented herein implies that algorithmic differences in state-of-the-art GANs become less relevant, as the computational budget increases. Furthermore, given a limited budget (say a month of compute-time), a "good" algorithm might be outperformed by a "bad" algorithm."

The downside, however, is that in 95% of cases the drawings still show errors. Some limbs are missing, faces are sometimes deformed and backgrounds blend into the characters here and there. So as such, these drawings are not yet suited to be published in a book or a newspaper.

Another major drawback is that the generated images seem to lack the variability that is present in the dataset. Let's compare this for a moment with StyleGAN2 trained on FFHQ as shown in [Karras et al., 2019]. The FFHQ model has learned to create images of new faces that look so good that it is hard to believe they are not pictures taken from existing individuals. In this experiment, however, training StyleGAN2 did not produce drawings depicting perfectly understandable new situations. An example will illustrate this flaw. In the dataset, there are drawings of Kinky & Cosy on the back of a camel in the desert. And there are also drawings of Kinky & Cosy on a bicycle on the street. A new image of Kinky & Cosy cycling in the desert would be an example of an interesting new synthesized drawing. The lack of such images is even more apparent when we generate interpolations in latent space (see section 5.6). The reason may be due to several issues. Overfitting of the discriminator is one of them. Excessive regularization of the generator is another. For the first, the solution is a larger augmentation and for the second, lower values for R_1 could help. But no miracles are to be expected as shown in section 5.5.

Another downside is the oversupply of walk-cycles and bicycle cycles. The cause, as mentioned earlier, is a weakness in the dataset. These animated cycles create a large concentration of nearly identical images. As a result, the data distribution exhibits too many peaks in a few limited places. A solution to this lies in striving for normal data distribution. This can be achieved by purifying the dataset so that only completely different images appear in it.

5.5 Hyperparameters

5.5.1 R_1 Regularization and ADA

[Karras et al., 2020] makes a suggestion concerning the choice of γ for the R_1 regularization: "Although the optimal γ does vary wildly, from 0.01 to 10, it seems to scale almost linearly with the resolution of the dataset. In practice, we have found that a good initial guess is given by $\gamma_0 = 0.0002 \times N/M$, where $N = w \times h$ is the

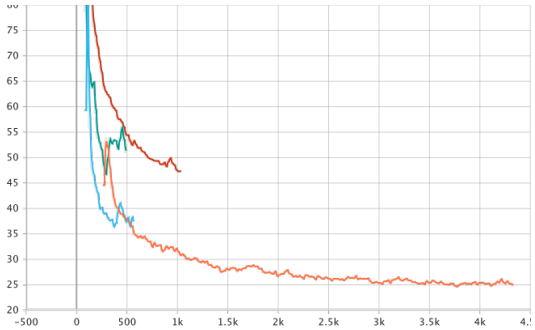


Figure 5.5: FID for different values of $\gamma = 0.5(green), 5(blue), 50(orange), 500(red)$.

number of pixels and M is the minibatch size. Nevertheless, the optimal value of γ tends to vary depending on the dataset, so we recommend experimenting with different values in the range $\gamma \in [\gamma_0/5, \gamma_0 \times 5]$." I reran the second experiment, each time with a different value for $\gamma \in \{0.5, 5, 50, 500\}$. The augmentation I chose was maximum. As figure 5.5 shows, 50 is indeed a good value. Lower values show earlier convergence at a higher value of FID. This produces models that give less good results. Indeed, more variation occurs at lower gamma values but it does not result in the hoped-for result. Instead of showing a bit more diversity of the dataset in the generated images, the images form a kind of freer interpretation of the cartoon characters. The shapes become a bit more elastic and everything is much less correct. Higher values also result in a higher value of FID.

5.5.2 Batch size

The literature mentions that the generation of high-quality images from complex datasets remains a challenge [Abdal et al., 2019b]. [Brock et al., 2018] proposed BigGAN and argued that the training of GANs benefits dramatically from large batch sizes: "[...] simply increasing the batch size [...] improves the state-of-the-art IS [...]. We conjecture that this is a result of each batch covering more modes, providing better gradients for both networks. One notable side effect of this scaling is that our models reach better final performance in fewer iterations, but become unstable and undergo complete training collapse." The batch size in the experiments I performed is however limited by the RAM size of the GPU (8Gb). At a resolution of 1.024^2 batch size had to be limited to 2 which is a disadvantage for diverse datasets.

5.6 The Latent Space

Exploring the latent space gives a better idea of the composition of the modeled data distribution. As a first step, generating an image with truncation = 0 is interesting. As explained in the paper that coined the term *truncation trick*: "Truncating a $\mathbf{z}$ vector by re-sampling the values with magnitude above a chosen threshold leads to

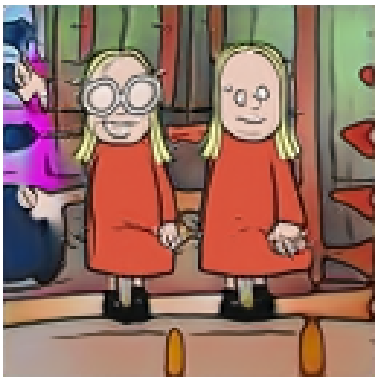


Figure 5.6: The mode of G's output distribution. (size 128×128 pixels)

improvement in individual sample quality at the cost of a reduction in overall sample variety. [...] As the threshold is reduced, and elements of $\mathbf{z}$ are truncated towards zero (the mode of the latent distribution), individual samples approach the mode of G's output distribution." [Marchesi, 2017] The seed no longer plays a role because all vectors that fall outside the normal distribution with standard deviation = 0, and they all are, are resampled. In other words, we always get the same image that is so to speak the average image of the modeled distribution (see figure 5.6). Oddly enough, this is not even a flawless image. But it is a very classic image, namely Kinky and Cosy standing side by side looking directly into the 'lens'. The background looks like a mixture of different backgrounds.

A subsequent experiment shows the progression of images when the truncation is incrementally increased. Up to 1 it falls within a normal distribution, after that it falls outside of it, so the images become more extreme, up to completely wrong (see figure 5.7).

Next, interpolations are of interest. Interpolations are possible in $\mathcal{Z}$ and in $\mathcal{W}$. But the latent space for G is simply $\mathcal{W}$. It is interesting to test both anyway. Visually, it appears that the difference between interpolations in $\mathcal{Z}$ and $\mathcal{W}$ is that fewer fast transformations occur in $\mathcal{W}$ than in $\mathcal{Z}$. This confirms the theory in [Karras et al., 2018]: "[...] a less curved latent space should result in perceptually smoother transition than a highly curved latent space." Because $\mathcal{W}$ is less warped than $\mathcal{Z}$, the perceptual distances are shorter.

Even better, instead of linear interpolations, I use quaternion spherical linear interpolations as explained in Section 3.5. This avoids sampling at low-probability regions of the latent space. As shown in figure 5.8, this interpolation loop is far from perfect. Because the radius of the loop is limited, not that much happens. But even then, there are many non-sensical images in it. Initially, Kinky and Cosy change places, and the background changes. But halfway through, all kinds of overlapping forms arise that don't depict anything interpretable. I ran this interpolation loop with the same parameters on StyleGAN2 FFHQ. This produces a series of realistic

Figure 5.7: Images generated with the same seed, but with **z** sampled from a truncated normal with increasing magnitude (from upper-left image to lower-right ranging for 0 to 20 with step 0.1)

error-free faces. Compare to that model, the conclusion is that there is still a lot of work to be done before we get similar results with comic drawings.

5.7 Meaningful Interpolations

Nevertheless, there are some bright spots. While experimenting, I came across two semantically meaningful interpolations. I found them accidentally in the many interpolations that I generated. Closed-form-factorization [Shen and Zhou, 2020] is a technique to find 'human readable' vectors, so I tried this first but without interesting results. The reason probably lies with the unstructured nature of the dataset. But they do exist, as figure 5.9 illustrates. We see an interpolation where Kinky's head is rotated about 90°. This is a particularly interesting discovery because these drawings are present to a much more limited extent in the dataset. In the design phase of a cartoon series production, the number of drawings is strictly limited due to budget restrictions. For Kinky & Cosy, the main characters were given only 5 drawings between a head looking $\frac{3}{4}$ left and $\frac{3}{4}$ right. (See appendix A). Of course, in-betweens were also created whenever the animation needed it to be smoother but

Figure 5.8: A circular walk through the latent space with random seed 67, truncation 0.5 and diameter 25

what we see here is a very fluid animation of a turning head. So this interpolation was performed in a semantically meaningful direction for a well-defined layer in the generator. This is due to the disentanglement of the latent space $\mathcal{W}$. As explained in [Karras et al., 2018]: "If a latent space is sufficiently disentangled, it should be possible to find direction vectors that consistently correspond to individual factors of variation." Another example of such meaningful interpolation is shown in figure 5.10. There, the legs of both characters are stretched fluidly in something resembling a slow-motion walk-cycle.

5.8 Projection

As explained in 2.2.7, projection means finding the latent vector $\mathbf{w}$ that generates a given image. This is an optimization problem whose result depends on the quality of the optimization algorithm, the trained model, and whether the image was in the training dataset or not. As an experiment, I performed pixel-based loss projection of three different images. A first image is an image generated by the model. That is the simplest test because an image that the model already created, should be able to effortlessly find its $\mathbf{w}$ vector. A second image is an image that is in the dataset

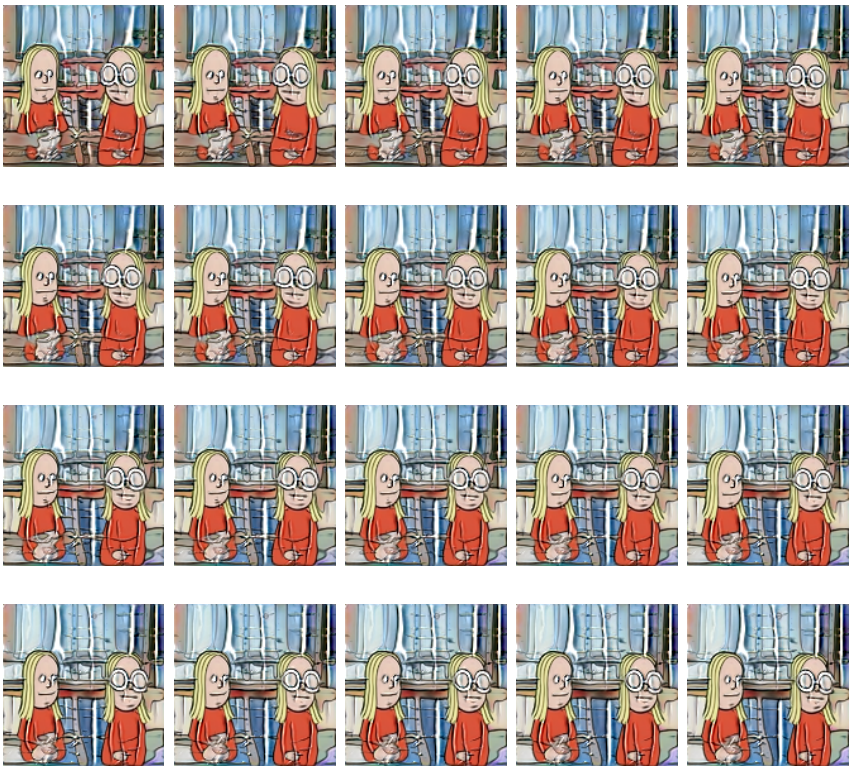


Figure 5.9: Example of a semantically meaningful direction in the latent space $\mathcal{W}$. Kinky (on the right) turns her head smoothly from $\frac{3}{4}$ left to $\frac{3}{4}$ right in this interpolation.

but that I didn't see any evidence of during the generations. A third projection was done based on an image that was not generated and was not in the dataset. Thus an entirely new image, but an image of Kinky & Cosy with different gestures that do not appear in the cartoons. Figure 5.11 shows the results. As expected the first image is projected almost flawlessly. The second projection succeeds nicely for the characters Kinky and Cosy but for the side characters depicted between them, the generation fails. I suspect that there are too few occurrences of this side character in the database so that no trace of it can be found in the modeled data distribution. The last image succeeds amazingly well. The reason, of course, is that it is not a very anomalous image. But still, the model manages to pull a kind of zero-shot creation out of its sleeve.

5.9 Network Blending

Network blending is the result of merging two different models, by splitting them at some level and combining the lower layers of one model with the higher layers of the other. The principle is similar to style transfer but at the network level. Although

Figure 5.10: This walk in the latent space $\mathcal{W}$ is also a walk on the sidewalk for the Kinky & Cosy. It is a semantically meaningful direction in $\mathcal{W}$ because both characters stretch their legs as part of smooth walk animation. Additionally, Kinky (on the left) turns her head towards 'the camera' and smiles.

this is not the subject of this thesis, it is interesting to check if a blended model coming from one trained at FFHQ and one trained at Kinky & Cosy gives interesting results. I expect that if the higher layers are derived from the Kinky & Cosy model, the generated images might have a style transfer look similar to animeGAN (see figure 2.2b). However, the result is disappointing (see figure 5.12). Presumably, the cause lies in the fact that the Kinky & Cosy images do not consist exclusively of centered and neatly cropped faces of the characters. Thus, there is no synergy with the FFHQ images. The red color of the dresses is picked up as face color, which is an undesirable result. The reverse blend did produce graphically intriguing images but they are not useful for this study.

5.10 The Pre-trained Model

I analyzed the influence of the dataset of the pre-trained model by testing four different pre-trained models (see figure A.3) on images of resolution 128^2. There

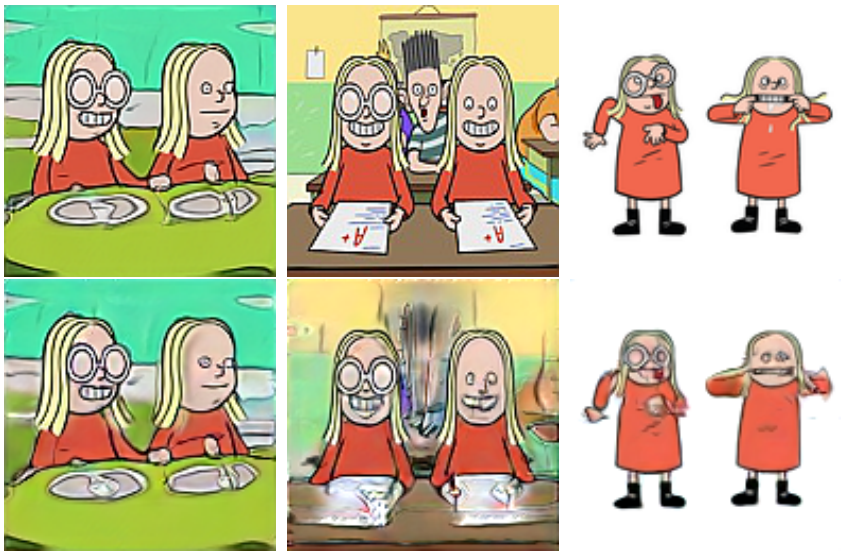


Figure 5.11: Projection of images in the latent space. Top: the source images, bottom: the targets. From left to right: a generated image, an image from the training set, and an image that is never seen by the model.

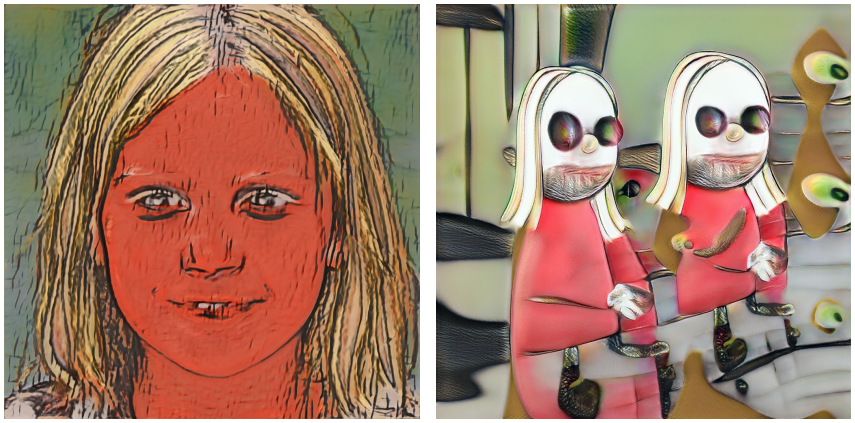


Figure 5.12: Generated images from blended models. Left: lower layers from FFHQ and higher layers from Kinky & Cosy with a split at resolution 128^2. Right: the reverse.

were no noticeable differences, neither in learning speed nor image quality. Only at higher resolutions, one can notice the different starting points in the first generated samples. But after a sufficient number of iterations, the difference disappears, and the models all train alike.

5.11 A Higher Resolution

For publication purposes, we need higher resolution images. StyleGAN's highest resolution is 1024^2. That is enough for publication but higher would be even better. In book printing, the drawing is often delivered in two separate files. One for the black line and the text and one for the colors. The black line then reaches a resolution that is at least double what StyleGAN can now handle. For publication on websites, the requirements are much less strict.

For the highest resolution of 1024^2 however, it proved to be harder to converge the model. When the hyperparameters are not correctly tuned, training results in images referring to the Kinky & Cosy style at 40 kimg, but then drift off and diverge. Presumably, the reason is as mentioned in [Karras et al., 2020]: "Transfer learning gives significantly better results than from-scratch training, and its success seems to depend primarily on the diversity of the source dataset, instead of the similarity between subjects. For example, FFHQ (human faces) can be trained equally well from CELEBA-HQ (human faces, low diversity) or LSUN DOG (more diverse). LSUN CAT, however, can only be trained from LSUN DOG, which has comparable diversity, but not from the less diverse datasets." It appeared that models with lower resolution do not as much suffer from this.

Training produces interesting images after the first few kimg. At a scale of 128^2, the first samples are just random gray noise. But at full scale, we get immediately interpretable results. For FFHQ as shown in figure 5.13 samples showed spooky red faces with grim smiles and blond striped hair. I hypothesize that the model picked up three striking characteristics from the Kinky & Cosy database, namely the dominant red color and the smile with clearly visible teeth, and the blond hair. These are features present in the low-resolution layers. [Karras et al., 2020] has shown that StyleGAN2 proceeds in the same way as the progressive growing model, that is, it will first adjust mainly the low resolution and afterward mainly the high resolutions. As a result, these are the first changes that become visible. Of course, training time increased significantly at a higher resolution on a laptop with 1 GPU. The results of incomplete convergence are comparable to those of lower resolution.

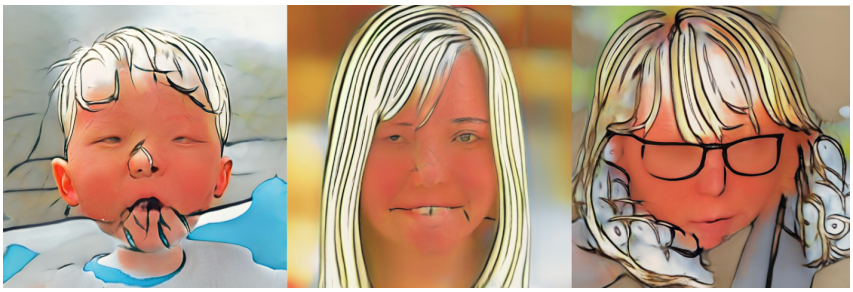


Figure 5.13: Fake images after 12kimg of transfer learning on model pre-trained on the FFHQ dataset.

5.12 Transfer Learning on Comic Dataset

As explained in Appendix A, cartoons are characterized by a fixed design of the characters. They are placed in a montage against a background and the position of their bodies is shaped as much as possible like a marionette. This is in stark contrast to the creation process of a comic strip, in which the drawing is made from scratch time and time again, resulting in all kinds of errors and deviations. The Kinky & Cosy comics enjoy a much greater degree of artistic freedom than the animated cartoons. Moreover, over the years, the comic strips of Kinky & Cosy have been subject to an evolution in style. Therefore, it seemed interesting to perform supplementary transfer learning on a small dataset derived from the comics (512 images).

Figure 5.14: Real images of Kinky & Cosy comic strips.

Figure 5.14 shows samples from this dataset. Figure 5.15 shows a sample of the fakes after convergence. What is striking is that many more incorrectly drawn figures appear. It seems as if the distorted drawings from the strip have 'confused' the model. In addition, patches of speech bubbles pop up here and there with non-sensical characters. One reason for this is undoubtedly the very limited size of the comic strips dataset. It would be interesting to repeat this experiment on a much larger dataset.

5.13 Conclusion

The results of StyleGAN2 are impressive. It is possible to generate clear drawings in the recognizable style of Kinky & Cosy. The number of errors is limited but too large to produce publishable material. Presumably, the cause lies in the too great diversity of the dataset. As mentioned in [Sauer et al., 2022]: "StyleGAN's performance severely degrades on large unstructured datasets [...]. StyleGAN was

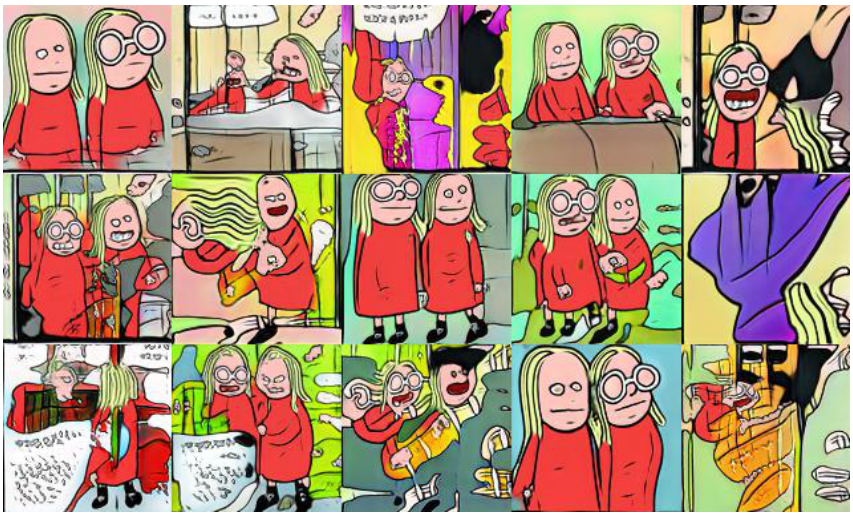


Figure 5.15: Fake images of Kinky & Cosy comic strips after transfer learning on the model pre-trained on the Kinky & Cosy animated cartoons.

designed for controllability; hence, prior works suspect its restrictive design to be unsuitable for diverse datasets." Thus, the search must continue toward new models and a modified dataset. The presence of semantically meaningful directions in latent space hints at the next step in this research. The StyleGAN improvements on latent space disentanglement allow for the exploration of single attributes of the dataset in an orthogonal way (i.e. without affecting other attributes). Provided training on a more coherent dataset, where only the characters are depicted full-scale and without overlap, it should be possible to learn these meaningful directions as described in [Abdal et al., 2019b], [Abdal et al., 2019a] and [Shen and Zhou, 2020].

Chapter 6

Conclusion

This research showed that with current state-of-the-art models such as StyleGAN2, style imitation of cartoon drawings is possible. Purely stylistically, there is a very good approximation. In terms of content, however, the variance remains too limited and the drawings still show errors. This has two causes: the quality of the dataset and the capabilities of the model. As such, these results are not suitable for publication. Moreover, we are still very far from the final moonshot: namely, the autonomous generation of fully finished comic strips. In the last few weeks before finishing this text, new text-to-image models have been popping up with increasing frequency. DALL-E 2 hit us all with amazement. But also VQ-GAN+CLIP, Stable Diffusion, Imagen, and others show impressive results. There are also very powerful text generation models, so making comics is something that will probably succeed before the end of this decade. One component of this is style imitation, which was the subject of this thesis. I hypothesize that comic panel generation will be done through a combination of composition and synthesis of characters, backgrounds, and text balloons with generative models. A process that approaches the actual work done by comic artists and which reduces the complexity of image generation for the models.

6.1 Future Work

The results of this work can be a benchmark for future analyses of other comic datasets. I hope that other researchers will turn to the large number of comics available in databases on computers at all kinds of comic publishers. Although, of course, it will be necessary to obtain the permission of the rights holders and publishers.

As for the dataset, possible future work is more rigorous preprocessing. In the case of Kinky & Cosy, a new dataset should be created that exhibits less intrinsic diversity. This dataset could consist of images where Kinky and Cosy only appear full body, neatly cropped from head to toe. So all close-ups and images where the characters are cut off by objects or characters in the foreground should be removed. Other models could be tested such as StyleGAN-XL [Sauer et al., 2022] which claim to deliver better results for unstructured datasets. Of course, unconditional image

generation is not an endpoint. If one wants to generate comics it is not enough to produce random drawings. The characters must be correct and depicted in the right situations. So the next step is to test conditional GANs. This of course assumes the availability of a labeled dataset. In the case of Kinky & Cosy, there are storyboards with specific descriptions of the image and corresponding time codes. This is valuable information to train models with text input. But even if the images are not stills from animations but panels from comic books, it is possible to obtain tags. The content of the text balloons can be used and if scenarios are available, they can be used for this purpose. I look forward to following this evolution and being a part of it.

Appendices

Appendix A

Experimental Set-up

A.1 The Dataset of Kinky & Cosy

From Kinky & Cosy, there are comics and animated cartoons. The cartoons were chosen to create a dataset. The reason for this is threefold. In the comics, there are word balloons, which is annoying. Moreover, the comics are drawn over a period of many years so the style has evolved. And the dataset of the comics is a factor of 10 smaller than the cartoons.

The cartoon series consists of 500 30-second episodes, which at a frame rate of 25 fps is the equivalent of 375.000 frames. It is 2D animation that is almost entirely hand-drawn and animated. It is important to take a brief look at the creation process of the cartoons because it helps to better understand the dataset. A cartoon is created by making a composition of background, objects, and characters scene by scene. Then the characters are animated so that they talk and move. Because everything is done manually, a large team of illustrators is involved and many hours of drawing work are performed. The production of the series Kinky & Cosy took $3,5$ years and was carried out in various studios in Europe, the US, and Asia. To prevent the budget from exploding there is an upper limit on the number of drawings (see table A.1). This means that savings are made on the number of characters, costumes, backgrounds etc... but much more importantly on the number of positions of bodies and expressions of faces. Or on the number of walk-cycles. When characters walk through the screen, we see in fact the same repetition of 12 images. When a character turns his head, we see the limited number of drawings provided in the design phase. Of course, the so-called 'in-betweens' are added when the animation required it to appear more fluid. But it is important to realize that the dataset contains a fair amount of redundancy.

To illustrate, figure A.1 shows an example of what is called a 'turn' in an animation movie. That is the number of frames in the design to turn the body of a character. In Kinky & Cosy a turn consists of 5 drawings between $\frac{3}{4}$ left and $\frac{3}{4}$ right. Therefore, it is interesting that StyleGAN2-ADA has modeled a completely smooth turn, as shown in figure 5.9.

Thus, when preprocessing the dataset for training GANs, it was necessary to filter

#	Type
380	Backgrounds
89	Characters
235	Costumes
1.435	Objects
89	Walk-Cycles

Table A.1: The number of drawings in the design phase.

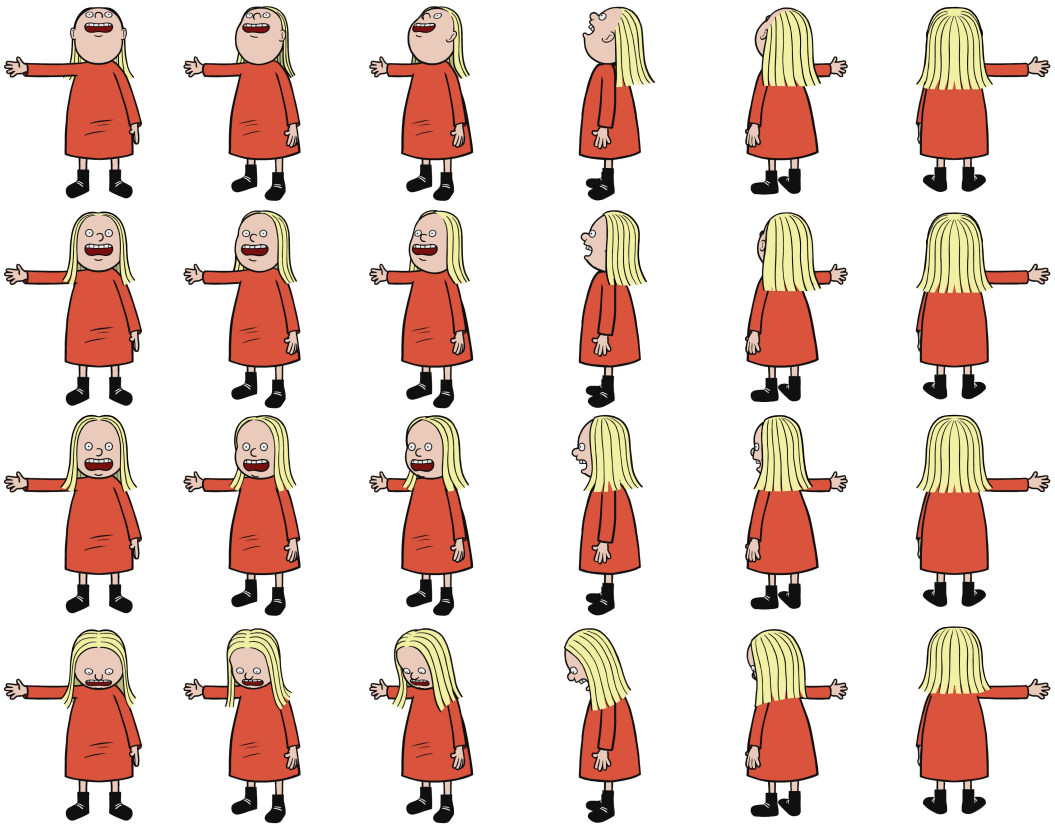


Figure A.1: In the design of the animation series, a turn of Cosy consists of a limited number of drawings.

these 375.000 images for redundancy. This was done while extracting the stills from the .mov files. Successive stills were compared with each other using the OpenCV `norm(image1,image2)` function. A threshold was set and the result was visually inspected. The main achievement of this filter is that so-called 'talking heads' were removed. There are an enormous number of scenes in which nothing happens at all because a character is just talking. Except for blinking eyes and moving mouths, everything is stationary. Moreover, 12 different mouths are sufficient to achieve lip synchronization. So there's a lot of redundancy there too. After this filter, the

images were cropped left and right so that they would be square. This produced an initial dataset of 56.234 images. Because it became apparent later during the experiments that the diversity of the dataset was too great to generate useful images, a new dataset was created. This was done by focusing on Kinky and Cosy in the images. A brute-force method was used to search for the color of the dress, skin, and hair. Then a square was cut out within which the character was present. This second dataset consists of 8.547 images which give significantly better results. However, this did not remove all redundancy from the dataset. The experiments show that. The reason is that walk-cycles and bicycle cycles have slipped through the net. These animations are characterized by sufficient differences between successive frames, but as a whole, they are still very monotonous.

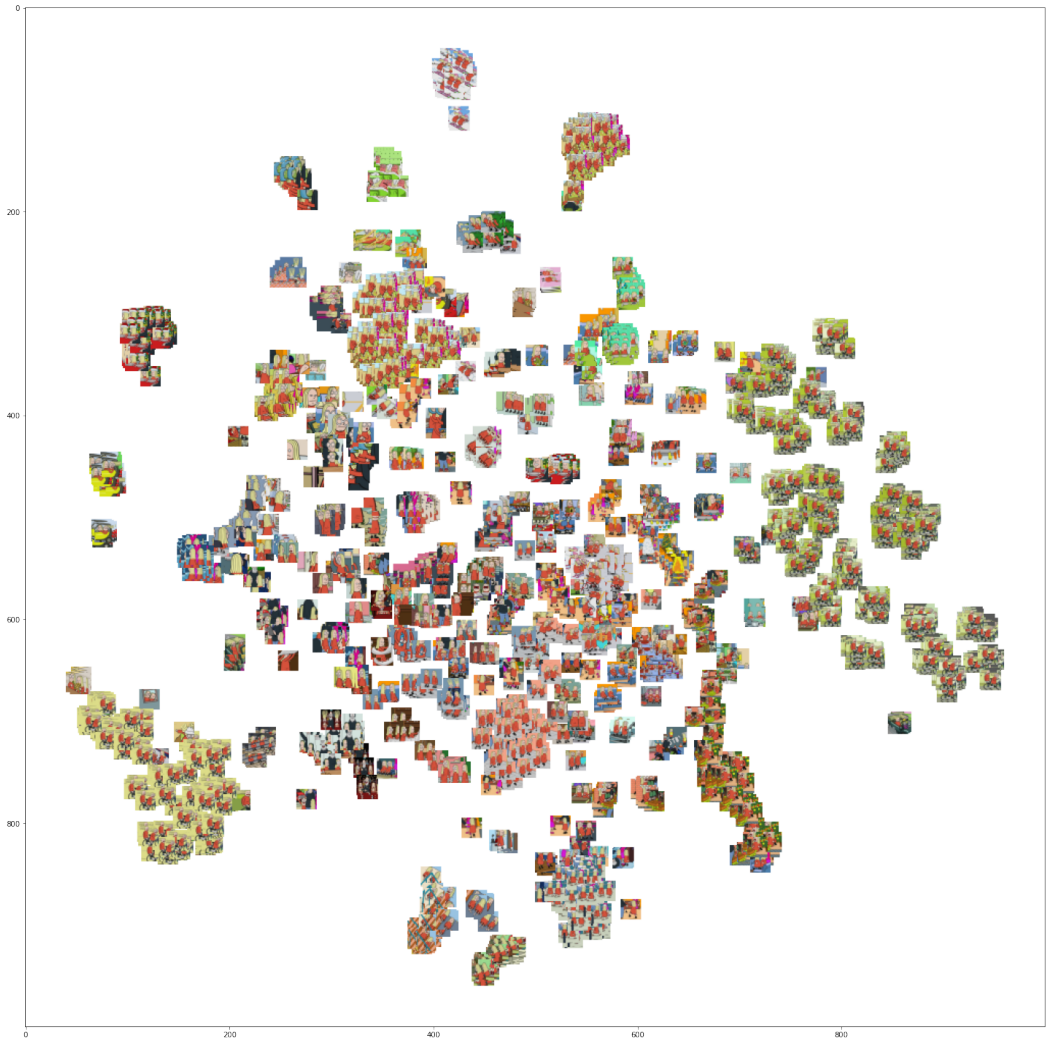


Figure A.2: A T-SNE plot of 2.000 consecutive samples from the dataset of images focusing on the main characters Kinky & Cosy.

Figure A.2 shows a T-SNE plot of the 2.000 consecutive samples from the dataset. The plot was made based on the features of a ResNet 101. From this, we can clearly see the cited problem. In addition to a cloud of various images, clear clusters of identical images occur. We recognize in it the walk-cycles, bicycle cycles, and other repetitive animation sequences such as dancing. In the data distributions, these repetitions of similar images are responsible for peaks of high probabilities for a limited number of images. As a result, disproportionate importance is given to these images which creates a distorted manifold. In future work, these peaks need to be removed from the dataset.

A.2 Pre-trained StyleGAN2 Models

The pre-trained models are taken from an overview on Github [Vasily, 2019]. I tested five models: Anime portraits (Danboru), My Little Pony (Derpibooru), Modern art (Frea Buckler artwork), Human faces (FFHQ), and Abstract art (WikiArt).

Figure A.3: A collection of pre-trained StyleGAN2 models trained on different datasets: Anime portraits, My Little Pony, Modern art, Human Faces, Abstract art.

Bibliography

[Abdal et al., 2019a] Abdal, R., Qin, Y., and Wonka, P. (2019a). Image2stylegan++: How to edit the embedded images? *CoRR*, abs/1911.11544.

[Abdal et al., 2019b] Abdal, R., Qin, Y., and Wonka, P. (2019b). Image2stylegan: How to embed images into the stylegan latent space? *CoRR*, abs/1904.03189.

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan.

[Augereau et al., 2018] Augereau, O., Iwata, M., and Kise, K. (2018). A survey of comics research in computer science. *Journal of Imaging*, 87(4).

[Brock et al., 2018] Brock, A., Donahue, J., and Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096.

[Cao et al., 2014] Cao, Y., Lau, R. W. H., and Chan, A. B. (2014). Look over here: Attention-directing composition of manga elements. *ACM Trans. Graph.*, 33(4).

[Chen et al., 2020] Chen, J., Liu, G., and Chen, X. (2020). Animegan: A novel lightweight gan for photo animation. In Li, K., Li, W., Wang, H., and Liu, Y., editors, *Artificial Intelligence Algorithms and Applications*, pages 242–256, Singapore. Springer Singapore.

[Creswell et al., 2018] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65.

[Detournay, 2022] Detournay, C.-L. (2022). La reprise de gaston lagaffe : imitation, hommage ou plagiat? *ActuaBD*.

[Domingos, 2012] Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

[Heusel et al., 2017] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium.

[Inkawhich, 2022] Inkawhich, N. (2022). DCGAN tutorial. https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html.

[Jin et al., 2017] Jin, Y., Zhang, J., Li, M., Tian, Y., Zhu, H., and Fang, Z. (2017). Towards the automatic anime characters creation with generative adversarial networks.

[Karras et al., 2017] Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation.

[Karras et al., 2020] Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. (2020). Training generative adversarial networks with limited data.

[Karras and Hellsten, 2021] Karras, T. and Hellsten, J. (2021). Stylegan2-ada — official pytorch implementation. https://github.com/NVlabs/stylegan2-ada-pytorch.

[Karras et al., 2018] Karras, T., Laine, S., and Aila, T. (2018). A style-based generator architecture for generative adversarial networks.

[Karras et al., 2019] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2019). Analyzing and improving the image quality of stylegan.

[Kodali et al., 2017] Kodali, N., Abernethy, J., Hays, J., and Kira, Z. (2017). On convergence and stability of gans.

[Liu et al., 2020] Liu, M.-Y., Huang, X., Yu, J., Wang, T.-C., and Mallya, A. (2020). Generative adversarial networks for image and video synthesis: Algorithms and applications.

[Long, 2020] Long, L. T. (2020). Photo animation for everyone. https://animegan.js.org/. Accessed: 2022-07-26.

[Lucic et al., 2017] Lucic, M., Kurach, K., Michalski, M., Gelly, S., and Bousquet, O. (2017). Are gans created equal? a large-scale study.

[Marchesi, 2017] Marchesi, M. (2017). Megapixel size image creation using generative adversarial networks.

[Mescheder et al., 2018] Mescheder, L., Geiger, A., and Nowozin, S. (2018). Which training methods for gans do actually converge?

[Morris, 2021] Morris, D. (2021). Synthesizing comics via conditional generative adversarial networks. http://resolver.tudelft.nl/uuid:2a21a50a-81c4-4d0d-9dbc-5dbce67f2936.

[Pan et al., 2019] Pan, Z., Yu, W., Yi, X., Khan, A., Yuan, F., and Zheng, Y. (2019). Recent progress on generative adversarial networks (gans): A survey. *IEEE Access*, 7:36322–36333.

[Proven-Bessel et al., 2021] Proven-Bessel, B., Zhao, Z., and Chen, L. Y. (2021). Comicgan: Text-to-comic generative adversarial network. *CoRR*, abs/2109.09120.

[Radford et al., 2015] Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks.

[Salimans et al., 2016] Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *CoRR*, abs/1606.03498.

[Saswat, 2020] Saswat, D. (2020). Wgan-pytorch. https://github.com/saswatpp/WGAN-pytorch.

[Sauer et al., 2022] Sauer, A., Schwarz, K., and Geiger, A. (2022). Stylegan-xl: Scaling stylegan to large diverse datasets.

[Shen and Zhou, 2020] Shen, Y. and Zhou, B. (2020). Closed-form factorization of latent semantics in gans. *CoRR*, abs/2007.06600.

[Vasily, 2019] Vasily, K. (2019). Awesome pretrained stylegan2. https://github.com/justinpinkney/awesome-pretrained-stylegan2.

[Veale, 2022] Veale, T. (2022). Two-fisted comics generation: Comics as a medium and as a representation for creative meaning. https://computationalcreativity.net/iccc22/wp-content/uploads/2022/06/ICCC-2022_19L_Veale.pdf.

[White, 2016] White, T. (2016). Sampling generative networks.

[Wiatrak et al., 2019] Wiatrak, M., Albrecht, S. V., and Nystrom, A. (2019). Stabilizing generative adversarial networks: A survey.

[Xu et al., 2017] Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., and He, X. (2017). Attngan: Fine-grained text to image generation with attentional generative adversarial networks.