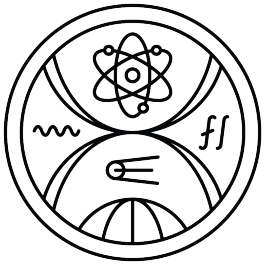
COMENIUS UNIVERSITY BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND
INFORMATICS

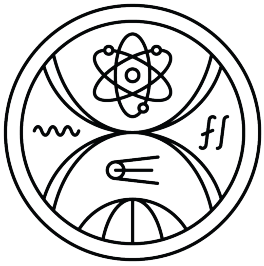


GENERATION OF SYNTHETIC 2D IMAGES OF TEETH

Diploma Thesis

2023 Bc. Dária Čárska

COMENIUS UNIVERSITY BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS



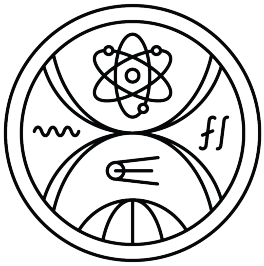
GENERATION OF SYNTHETIC 2D IMAGES OF TEETH

Diploma Thesis

Study programme:	mAIN/k - Applied Computer Science (Conversion Programme)
Field of Study:	Computer Science
Department:	FMFI.KAI Departement of Applied Informatics
Supervisor:	doc. RNDr. Martin Madaras, PhD.

Bratislava, 2023 Bc. Dária Čárska

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



GENEROVANIE SYNTETICKÝCH 2D OBRÁZKOV ZUBOV

Diplomová práca

Študijný program:	mAIN/k - Aplikovaná informatika (konverzný program)
Študijný odbor:	9.2.9 Aplikovaná informatika, 18. Informatika
Školiace pracovisko:	Katedra aplikovanej informatiky
Školiteľ:	doc. RNDr. Martin Madaras, PhD.

Bratislava, 2023 Bc. Dária Čárska

Comenius University Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname: Bc. Dária Čárska
Study programme: Applied Computer Science (Conversion Programme) (Single degree study, master II. deg., full time form)
Field of Study: Computer Science
Type of Thesis: Diploma Thesis
Language of Thesis: English
Secondary language: Slovak

Title: Generation of Synthetic 2D Images of Teeth

Annotation: Robust training of neural networks is based on access to the training datasets. A convolutional neural network can be trained in a deep learning manner using synthetic or real annotated data to perform instance segmentation of objects or advanced analysis of object properties and relationships between object instances. If there is no available real annotated training dataset a synthetic one can be used. Through a review of existing literature and a thorough investigation of state-of-the-art methodologies, this thesis aims to explore and propose techniques that leverage deep learning and computer vision to generate realistic and high-quality dental images. The synthesis of dental images plays a crucial role in various domains, including dental education, data augmentation for deep learning algorithms, and the development of computer-aided diagnosis systems. The proposed models will be evaluated using quantitative and qualitative measures to assess their effectiveness and compare them against existing methods.

Aim:
- Study relevant papers concerning the synthetic data generation
- Study relevant papers concerning convolutional neural networks, deep learning techniques for image processing, and generative adversarial networks for images
- Propose a system for synthetic teeth data generation
- Generate synthetic data of human teeth and focus on quantitative and qualitative evaluation and thesis writing

Literature:
Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. “Progressive growing of gans for improved quality, stability, and variation.” CoRR, abs/1710.10196, 2017

Kazuma Kokomoto, Rena Okawa, Kazuhiko Nakano, and Kazunori Nozaki. “Intraoral image generation by progressive growing of generative adversarial network and evaluation of generated image quality by dentists.” Scientific reports, 11(1):1–10, 2021

Tero Karras, Samuli Laine, and Timo Aila. “A style-based generator architecture for generative adversarial networks.” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition pages 4401–4410, 2019

Keywords: Synthetic data, data generation, parametric model, neural network, instance segmentation

Comenius University Bratislava
Faculty of Mathematics, Physics and Informatics

Supervisor:	doc. RNDr. Martin Madaras, PhD.
Department:	FMFI.KAI - Department of Applied Informatics
Head of department:	doc. RNDr. Tatiana Jajcayová, PhD.
Assigned:	20.10.2021
Approved:	13.05.2022

prof. RNDr. Roman Ďurikovič, PhD.
Guarantor of Study Programme

.....
Student

.....
Supervisor

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Dária Čárska
Študijný program: aplikovaná informatika (konverzný program)
(Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Generation of Synthetic 2D Images of Teeth
Generovanie syntetických 2D obrázkov zubov

Anotácia: Robustné trénovanie neurónových sietí je založené na prístupe k trénovaciemu datasetu. Konvolučná neurónová sieť môže byť natrénovaná metódami hlbokého učenia s použitím syntetických alebo reálnych anotovaných dát na úlohy inštančnej segmentácie alebo pokročilej analýzy vlastností objektov a vzťahov medzi objektami. Pokiaľ nie je dostupný reálny anotovaný trénovací dataset, tak vieme použiť syntetické dáta. Cieľom tejto práce je prostredníctvom prehľadu existujúcej literatúry a dôkladného preskúmania najmodernejších metodík skúmať a navrhovať techniky, ktoré využívajú hlboké učenie a počítačové videnie na generovanie realistických a kvalitných obrázkov zubov. Syntetické generovanie obrázkov zubov zohráva dôležitú úlohu v rôznych oblastiach, vrátane edukácie študentov zubného lekárstva, augmentácie dát pre algoritmy hlbokého učenia a vývoja systémov počítačom podporovanej diagnostiky stavu chrupu. Navrhované modely budú hodnotené pomocou kvantitatívnych a kvalitatívnych ukazovateľov, aby sa posúdila ich účinnosť a budú porovnané s existujúcimi metódami.

Cieľ:
- Naštudovať relevantné články ku generovaniu syntetických dát
- Naštudovať relevatné články ku konvolučným neurónovým sieťam, spracovaniu obrazu pomocou hlbokého učenia a GAN pre generovanie obrázkov
- Navrhnúť systém na generovanie syntetických dát
- Vygenerovať syntetické obrázky ľudských zubov a zamerať sa na kvantitatívne a kvalitatívne vyhodnotenie a písanie práce

Literatúra:
Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. “Progressive growing of gans for improved quality, stability, and variation.” CoRR, abs/1710.10196, 2017

Kazuma Kokomoto, Rena Okawa, Kazuhiko Nakano, and Kazunori Nozaki. “Intraoral image generation by progressive growing of generative adversarial network and evaluation of generated image quality by dentists.” Scientific reports, 11(1):1–10, 2021

Tero Karras, Samuli Laine, and Timo Aila. “A style-based generator architecture for generative adversarial networks.” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition pages 4401–4410, 2019

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Kľúčové slová: Syntetické dáta, generovanie dát, parametrický model, neurónové siete, inštančná segmentácia

Vedúci: doc. RNDr. Martin Madaras, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: doc. RNDr. Tatiana Jajcayová, PhD.

Dátum zadania: 20.10.2021

Dátum schválenia: 13.05.2022 prof. RNDr. Roman Ďurikovič, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Čestne prehlasujem, že túto diplomovú prácu som vypracovala samostatne len s použitím uvedenej literatúry a za pomoci konzultácií s mojim školiteľom.

Bratislava, 2023 .....

Bc. Dária Čárska

Acknowledgment: I would like to thank my supervisor doc. RNDr. Martin Madaras, PhD. for the guidance and invaluable insights throughout the elaboration of this diploma thesis. I would also like to express my gratitude to my family and especially my fiancé for their support.

Abstract

In recent years, due to the development of neural network architectures, algorithms, and hardware, many models have been implemented, helping to solve real-world tasks. Generative models, such as the renowned Generative Adversarial Network (GAN), have proven to be helpful in many domains, being able to produce synthetic data unrecognizable from real samples. The generated images of teeth have potential applicative usage in dental education and as explanatory materials for patients with different teeth conditions, as well as in further research without any privacy restrictions. In this thesis, we trained several famous GAN architectures to synthesize pictures of teeth from different views and let dental experts compare synthetic data with real photos. We describe the dataset used, data preprocessing, network training, as well as the design of each architecture.

Key words: Synthetic data, data generation, parametric model, neural network, instance segmentation

Abstrakt

V posledných rokoch našli neurónové siete uplatnenie v mnohých odvetviach vďaka vývoju nových architektúr, algoritmov a v neposlednom rade hardvéru. Generatívne modely, ako napríklad Generatívne Súperiace Siete (angl. Generative Adversarial Networks - GANs) sú nepostrádateľné v mnohých doménach vďaka ich schopnosti generovať nerozlíšiteľné syntetické dáta od reality. Potenciál využitia syntetických obrázkov zubov je najmä v zubárskom odvetví ako výukový materiál pre študentov zubného lekárstva, ktorí sa počas štúdia musia oboznámiť s rôznymi aberáciami chrupu a taktiež vo vedeckom prostredí, kde obrázky zubov môžu slúžiť pre ďalší výskum bez porušenia súkromia práv dotknutých osôb. V tejto diplomovej práci je opísané trénovanie viacerých populárnych GAN architektúr prispôsobených na generovanie syntetických obrázkov zubov z rôznych pohľadov. Podrobne je opísaná príprava dát, trénovanie sietí a dizajn každej architektúry. V poslednej kapitole opisujeme vyhodnotenie syntetických obrázkov vygenerovaných jednotlivými sieťami prostredníctvom anonymného formuláru expertmi z oblasti zubného lekárstva.

Kľúčové slová: Syntetické dáta, generovanie dát, parametrický model, neurónové siete, inštančná segmentácia

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Motivation

Nowadays, computer science offers many powerful tools which are capable of simplifying everyday work as well as personal lives of people. The automation of many processes has helped to improve efficiency and reduce human intervention in plenty of industry branches. However, the advent of machine learning, particularly deep learning, has brought remarkable advancements in various domains. Machine learning algorithms are able to analyze vast amounts of data, discover patterns, group similar data, predict outputs or classify data. These abilities have brought new opportunities for the processing of data and addressing complex problems which could not be previously overcome.

One of the areas in machine learning, which has become popular and well studied in recent research, is the image synthesis. Deep learning models perform very reliably in generating synthetic images and people often find it challenging to differentiate between real and synthetic images.

Lack of data is one of the main obstacles when trying to build a robust

model for completing a specific task. Therefore, producing synthetic samples can considerably enlarge the training dataset size and enables to effectively train a model.

1.2 Problem Definition

One of the challenges in the field of dental imaging is the limited availability of diverse and comprehensive datasets. Obtaining a large dataset of annotated teeth images is often costly and time-consuming, making it difficult to train robust and accurate machine learning models for various dental applications.

The main objective of this thesis is to address the problem of synthesizing high-quality dental images, specifically focusing on teeth, using Generative Adversarial Networks (GANs). Training a GAN on a limited dataset of real teeth images allows to generate a vast collection of synthetic dental images that closely resemble real teeth in terms of morphology, textures, and anatomical variations.

The generated synthetic dental images have the potential to serve several purposes in dentistry and oral healthcare. They can facilitate research and development by providing a virtually unlimited source of diverse dental images for testing and validation of dental algorithms and systems. Additionally, the synthetic dataset can be used to augment the limited real data, thus improving the performance of dental image classification, segmentation, and other image-based tasks.

1.3 Dataset

It is a common problem to find a dataset containing sufficient number of samples to train a machine learning model. The lack of data is especially present in the medicine research. There are legal and ethical limitations when sharing medicine data because of patient privacy. In this thesis, we decided to work with photos of human teeth which we used as input for the training of different generative models. After long research, we found an appropriate dataset, collected and downloaded altogether 21,242 available images of teeth from the Invisalign online database of treatments[1]. Invisalign is a company which produces clear, removable aligners that are used to straighten teeth of patients. The aligners are custom-made for each patient using 3D computer imaging technology. The patient wears a series of aligners, each for about one to two weeks, gradually moving the teeth into their desired position. In the Invisalign gallery, there is a treatment web page for each patient providing data, that is submitted by a dentist and consists of:

- Treatment info including patient information, diagnosis and treatment plan
- Intraoral photos before, optionally during and after the process of treatment
- Panoramic and cephalometric radiographs
- Treatment plan in form of a video simulation
- Treatment summary containing information about the treatment including number of aligners for maxilla and mandible, total treatment time, retention and achieved results

[1] https://global.invisalignгallery.com

Figure 1.1: Real images of all views (front, upper, lower, left, and right).

The teeth photos are made from various views before, during and after the treatment. Teeth are photographed from the front, left and right side and there are also photos of the upper and lower dental arches. We collected images of 1,721 patients from all five views available on the Invisalign webpage. Several example images from the dataset are shown in Figure 1.1.

Chapter 2

Theoretical Background

In this chapter, we are going to explain the theory and provide information about concepts which are the building blocks for the practical part of this thesis.

2.1 Machine Learning

Finding patterns is the ultimate activity which has shaped mankind and allowed it to progress in any discipline. Whether its medicine, astronomy, economy, developing powerful modelling techniques describing laws of nature is the ultimate goal.

Where human capabilities seize to suffice, computing machines excel at crunching through tons of data. Whether it is as simple as linear regression where one expects simple linear relationship between the predictive variables and the outcome, to large multi-layer neural networks that are able to approximate function of any kind.

2.1.1 Types of Machine Learning Algorithms

Finding patterns in data does not necessarily equate to predicting an outcome from a fixed set of variables. In this subchapter, we discuss three machine-learning approaches, all quite functionally different, nevertheless still achieving the same goal: searching for patterns.

Supervised Learning

Machine-learning in its simplest form is simply predicting an outcome (e.g. weight) from variables which we believe are somewhat associated with changes in the outcome (e.g. age, height, etc.).

The choice of an algorithm then reflects our expectations about the complexity of the relationship. Selection of a model is also a crucial point in the design of an experiment, as more simple models, such as linear regression might not perform as well as more complex models at the cost of higher parameter variability when training new models. On the other hand, simpler models tend to show higher systematic bias when predicting complex relationships. The bias-variance ratio should always be considered during the initial stages of an experiment. On top of that, complex models tend to be harder to interpret and thus are rarely used in critical areas, such as medicine.

In supervised learning the aim is to solve one of two problems: regression, where the outcome is continuous, such as weight, or classification where the outcome is categorical, such as sex. Each of these scenarios requires special care, nevertheless the algorithms can usually be well used in both cases.

For example, in neural networks the only thing that changes is the activation of the last neuron(s), so that the outcome is a probability, i.e. it falls in 0 to 1 range [12].

Unsupervised Learning

In the unsupervised learning, the goal is to learn patterns in data in the absence of labels. This includes clustering algorithms, anomaly detection or general data analysis. In many scenarios, there are only limited labeled data points while obtaining unlabeled data is easier. In this case, the combination of supervised and unsupervised methods tends to lead to more robust models (semi-supervised learning) [12].

Reinforcement Learning

Reinforcement learning describes scenario where an agent learns from previous experiences and reinforces good habits while avoiding punishable behaviors. This has been used in many recently popular algorithms, such as Deepminds AlphaGo [25], in self driving cars or teaching computer to play video games. The machine usually starts from scratch and explores the environment while being rewarded or punished for its actions. After many iterations, the machine is usually able to navigate the environment while maximizing the reward [1].

2.1.2 Neural Networks

Neural networks are relatively simple and straightforward model to understand, although extremely versatile and adjustable. With the innovations and progress in graphics card industry, neural networks became the most popular models for solving any task.

Their potential and the strength of using deep layer networks has been undoubtedly shown in 2012 in an image net competition, where a team constructed and trained a neural network called AlexNet which has won the

competition by great margin [19].

Another example is the Alphafold network. In 2018, a Deepminds team submitted their protein structure predictions and won the CASP (Critical Assessment of protein Structure Prediction) competition, again by great margin [24].

Neurons, weights and activation functions

Neural network is composed of interconnected units called neurons. The connections between the neurons are called weights. Similar to biological neurons, where learning results in better signal conductivity due to more myelinated axons, in neural networks the learning process simply means finding the right strengths of connections, i.e. weights.

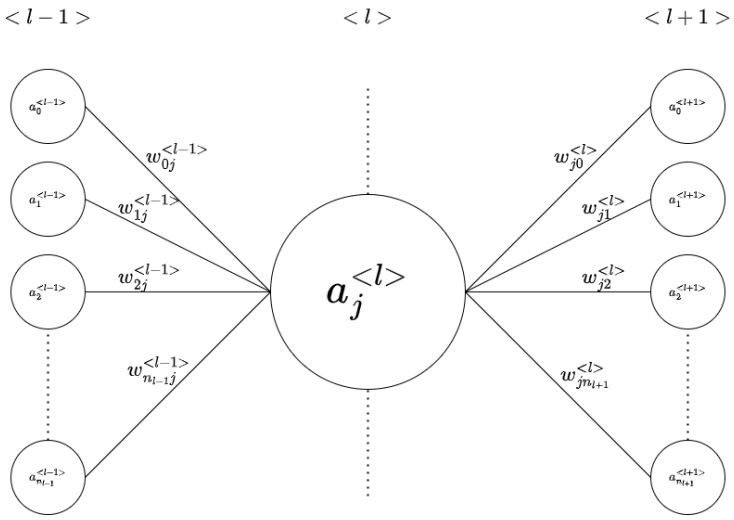


Figure 2.1: Schema of a neuron and weights.

The activation value of a neuron is a sum of all incoming neurons with corresponding activations from the last layer. Formally:

$$a_j^{<l>} = \Sigma_{i=1 \to n_{l-1}} w_{ij}^{<l-1>} \cdot a_i^{<l-1>}, \tag{2.1}$$

where $a_j^{<l>}$ is the activation of jth neuron in lth layer in the network and $w_{ij}^{<l-1>}$ is the weight connecting ith neuron in $l-1$th layer with jth neuron in lth layer.

The activation function is some non-linear function, such as tanh, sigmoid or relu [1].

2.1.3 Training

The training of a neural network consists of repetitive forward passes through the network and weight adjusting passes backwards. The technique used is called gradient descent, where parameters of a model are always changed in the direction of steepest gradient decrease. In this way, if the steps are reasonably small, the neural network ultimately reaches some local optimum of the loss function. Reaching global optimum is not computationally feasible in large neural networks, however some tricks such as momentum might help escape some local optima [1].

Loss Functions

In order for the network to learn, it must be able to quantify how far from the reality its predictions are. In regression setting, this is usually achieved by calculating the mean squared error:

$$L = \frac{1}{n}\Sigma_n(y_i - \hat{y}_i)^2. \tag{2.2}$$

No activation function is usually applied to the last neuron in regression setting. However, this is not true for classification, where it is desired state to have probabilities for each class. Thus, before calculating the loss, a normalization function such as softmax activation function is applied to each

neuron in the last layer:

$$a_i^{l_m} = \frac{e^{z_i^L}}{\Sigma_j e^{z_j^L}}, \tag{2.3}$$

where z is the pre-activation value of a neuron (weighted sum of activations from last layer) and l_m denotes the final layer. This ensures that values of neurons in the last layer are between 0 and 1 range and sum up to 1.

Finally, the loss is calculated for each output neuron, usually cross entropy:

$$L_i = -y_i \cdot \log \hat{y}_i - (1 - y_i) \cdot \log 1 - \hat{y}_i. \tag{2.4}$$

In this way, the loss will be small if a neuron fires up when it is supposed to (i.e. the correct neuron outputs high probability) and conversely the loss will be high for neurons which are off the target.

Finally, since the value of each output neuron is a function of each weight in previous layers, one can find a partial derivative of the loss according to each weight, which tells how the weight should be adjusted so that the loss in next run is smaller [1].

2.2 Convolutional Neural Networks

In its simplest form, feed forward neural networks work great for datasets with unrelated attributes. In cases where spatial information is important, such as in image recognition, convolutional neural networks became the go-to solution.

The convolution operation applies a kernel of predefined dimensions (e.g. 2x2x3 in the Figure 2.2) to the input and computes the weighted sum of the

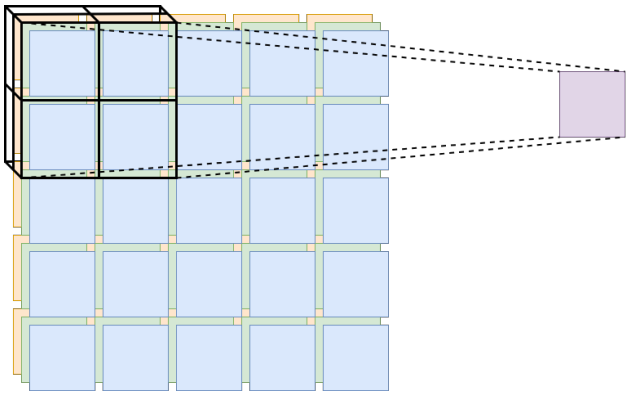


Figure 2.2: Illustrated convolution operation.

kernel values (weights) with the input values. The kernel then traverses over the entire input, producing a new layer of the network. Usually, in order to maintain the dimensions of the previous layer the input is padded, although many times dimension reduction is desired. Historically, to achieve this, pooling operation has been used as well. In this case, some simple operation is applied to the input such as maximum, average, median, etc. However, since the network is not learning at this step, convolution generally performs better with large datasets [1].

Chapter 3

Generative Models

The aim of generative modeling is to produce synthetic data unrecognizable from real samples. The model learns to represent the underlying distribution of the input data and afterwards is able to create new data instances that resemble the training data. There are plenty of tasks and applications where synthetic data can be useful, such as synthetic image, text, audio or video generation. Using synthetic data requires much less memory than storing real datasets and generated samples can be used for data augmentation in cases where the provided data is limited. In this chapter, we describe more in detail two types of generative models - a generative adversarial network and a variational autoencoder, which are commonly used for generative tasks.

3.1 Variational Autoencoder

Autoencoder is a popular neural network architecture generally used for dimensionality reduction, similar to PCA. It consists of an encoder and decoder parts with a typical symmetric architecture with a bottleneck in the middle. The goal is to extract as much information from the input using less features

and be able to reconstruct it with as minimal loss as possible.

Variational Autoencoders (VAE) is a special type of this architecture, where the latent space contains encodings of the input as a distributions rather than a single values. The decoder then samples points from distributions in the latent space and generates new data. Variational autoencoder is thus a regularised version of classical autoencoder.

VAE can be used for generating any kinds of data, however they tend to perform slightly worse than generative adversarial networks due to the strict regularization of the latent space [6].

3.2 Generative Adversarial Network

Generative adversarial network (GAN) [6, 1] is a type of neural network that is used for generative tasks, such as image generation, text generation and audio synthesis. GANs consist of two main components: a generator network and a discriminator network. The generator network is responsible for generating new data samples, while the discriminator network is responsible for distinguishing the generated samples from real samples. Generative adversarial networks are based on a game theoretic scenario in which the generator network must compete against an adversary. The GAN architecture is illustrated in Figure 3.1.

3.2.1 Generator

The aim of the generator part of a GAN is to learn to generate synthetic believable data unrecognizable by the discriminator from real world samples. As an input, the generator takes a fixed-sized p-dimensional vector and feeds it through the neural network architecture yielding a d-dimensional vector

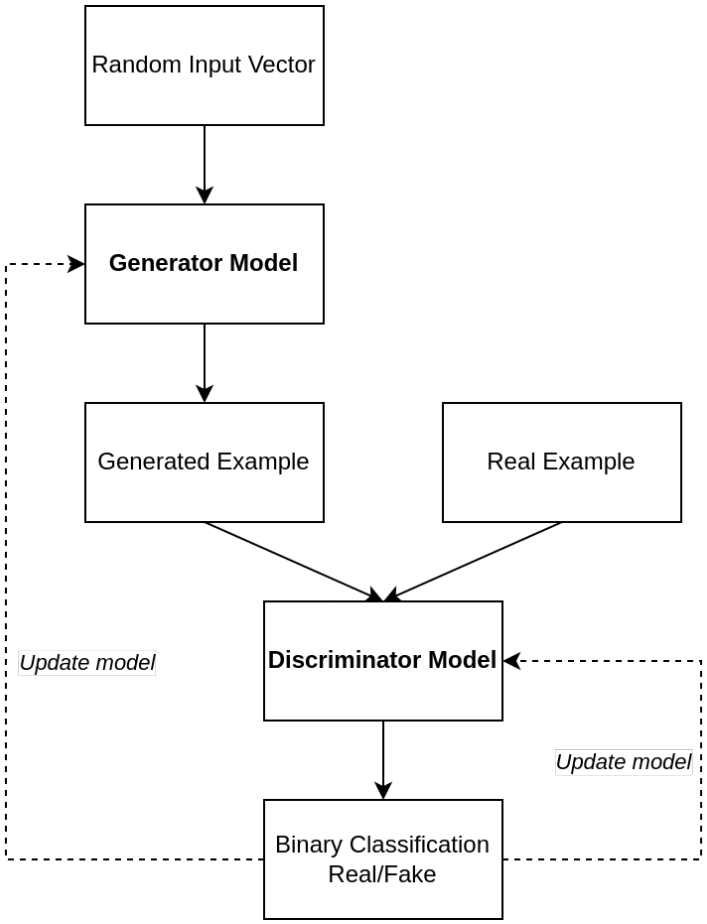


Figure 3.1: Diagram of the Generative Adversarial Network model architecture [4].

representing datapoint coming from a similar distribution as the samples from the real world dataset. The generator optimizes its parameters based on feedback provided by the discriminator with the aim to fool the discriminator into thinking that the datapoint is real.

3.2.2 Discriminator

Discriminators goal is to distinguish between synthetic data produced by the generator from real ones. Thus it is a simple binary classifier. The choice

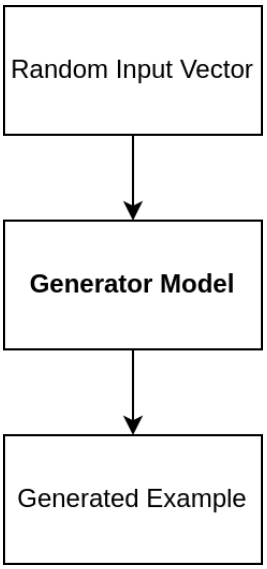


Figure 3.2: Scheme of the Generator model [4].

of architecture depends on the task at hand, for example a Convolutional Network for images. As an input it takes a d-dimensional vector either from a dataset of real samples or the generators output. The output is a probability, i.e. number between 0 and 1.

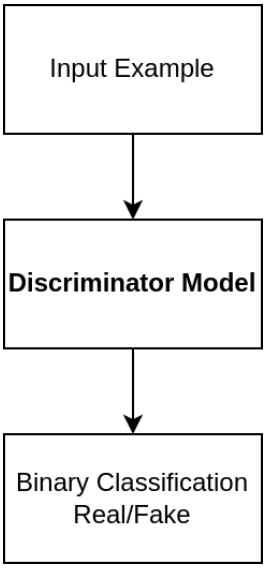


Figure 3.3: Scheme of the Discriminator model [4].

3.2.3 Training

The generator and discriminator networks engage in an adversarial training process, wherein the generator creates samples with an aim to deceive the discriminator, while the discriminator strives to accurately differentiate between the generated and real samples. This iterative process persists until the generator achieves the ability to produce samples that are practically indistinguishable from real ones.

The training of GANs [1] includes two primary objectives: First, it involves finding the discriminator's parameters that maximize its accuracy in classifying samples. Second, it aims to find the generator's parameters that fool the discriminator. This training process proceeds by iteratively updating the parameters of both the generator and the discriminator networks.

For the discriminator, its objective is to correctly classify real examples by assigning them a label of 1, and to identify synthetically generated examples by assigning them a label of 0. Conversely, the generator's goal is to create examples that effectively deceive the discriminator, prompting the discriminator network to label these artificially generated samples as 1.

Let R_m denote a set of m randomly sampled examples from the real dataset, and S_m represent a collection of m synthetic samples generated using the generator network. To create these synthetic samples, we start by generating a set N_m of p-dimensional noise samples $z_1, z_2, \ldots, z_m$. These random noise samples are then fed as inputs to the generator, resulting in the synthetic data samples $S_m = G(z_1), G(z_2), \ldots, G(z_m)$.

The objective function J_D for the discriminator is formulated as follows [1]:

$$maximize_D J_D = \sum_{x \in R_m} \log [D(x)] + \sum_{x \in S_m} \log [1 - D(x)]. \tag{3.1}$$

The objective function achieves its maximum value when the real samples R_m are classified correctly as 1, and at the same time, the synthetic examples S_m are classified accurately as 0.

Next, we establish the objective function for the generator, which aims to fool the discriminator. Unlike the discriminator, the generator is solely concerned with the samples it generates and does not take into account the real examples. By producing a set of m synthetic samples, S_m, the generator strives to ensure that the discriminator perceives these samples as authentic. Consequently, the generator's objective function, denoted as J_G, seeks to minimize the likelihood of these samples being identified as synthetic. This leads to the formulation of the following optimization problem [1]:

$$minimize_G J_G = \sum_{x \in S_m} \log [1 - D(x)] = \sum_{z \in N_m} \log [1 - D(G(z))]. \tag{3.2}$$

The primary aim of minimizing this objective function is to cause the synthetic examples S_m to be mistakenly classified as real ones, i.e. assigned a label of 1. By doing so, we effectively strive to learn the generator's parameters in a manner that deceives the discriminator into correctly classifying the synthetic examples as real samples from the dataset.

Alternatively, we can use a different objective function for the generator: maximizing $\log [D(x)]$ for each $x \in S_m$, rather than minimizing $\log [1 - D(x)]$. This alternative objective function sometimes proves more effective, especially during the early stages of optimization. By utilizing this modified minimax loss, we can prevent the training process from getting stuck in the

initial phases when the discriminator can easily differentiate between real and synthetic samples.

Hence, the optimization problem can be expressed as a minimax game over J_D. It is worth noting that maximizing J_G with respect to various parameter choices for the generator G is equivalent to maximizing J_D, as $J_D - J_G$ does not involve any of the generator's parameters. Consequently, we define an optimization problem with both the generator and discriminator networks, as follows [1]:

$$Minimize_G Maximize_D J_D. \tag{3.3}$$

The result of such optimization is a saddle point of the optimization problem.

The generator and the discriminator undergo different training processes. Stochastic gradient ascent is employed for learning the parameters of the discriminator, while stochastic gradient descent is utilized for learning the parameters of the generator. The gradient update steps are alternated between the generator and the discriminator. As a result, the GAN training progresses through alternating periods:

1. The discriminator trains selected number of epochs.

2. The generator trains for selected epochs.

Steps 1 and 2 are repeated until desirable outcome is achieved.

During GAN training, for each step of the generator, k update steps are performed for the discriminator. Typically, the value of k is kept small (less than 5), although it is also possible to use $k = 1$.

During the discriminator training phase, we keep the generator constant. The discriminator aims to distinguish real data from fake, and in doing so, it

learns how to identify the generator's flaws. This task differs for a thoroughly trained generator, which produces more coherent output, compared to an untrained generator that generates random samples.

Similarly, during the generator training phase, we keep the discriminator constant. Otherwise, the generator would be attempting to produce samples to match a continually changing discriminator, making it difficult to achieve convergence.

This iterative process of updating the parameters of both networks continues until convergence is reached, specifically until a Nash equilibrium is achieved. At this stage, the discriminator becomes incapable of differentiating between real and synthetic examples. Once the training is complete, the generator model is preserved and utilized to generate new samples for further use.

During the training process, certain factors require careful consideration. Firstly, if the generator undergoes extensive training without updating the discriminator, it may result in the generator repeatedly producing highly similar samples. This lack of diversity can be problematic, which is why simultaneous training of the generator and discriminator with interleaving is employed.

Secondly, in the early iterations, the generator might produce subpar samples, leading to low values for $D(x)$ (close to 0). Consequently, the loss function approaches 0, and its gradient becomes low. This saturation effect slows down the training of the generator's parameters. To address this, an alternative approach is sometimes used during the initial stages of training, where instead of minimizing $\log[1 - D(x)]$, the objective is to maximize $\log[D(x)]$. Although heuristically motivated, this adjustment tends to work well in practice, particularly when the discriminator rejects all generated

samples during early training.

3.2.4 Common Problems

There are two major problems that are common when training GAN models: mode collapse and convergence failure.

Mode Collapse

A mode collapse [4] occurs when a generator model is able to produce only one or a small subset of distinct outcomes, known as modes. In the context of a GAN, it is desirable to have a wide variety of outputs, such as different pictures of teeth, possibly with believable aberrations. However, in the case of a mode failure, the generator's vast input latent space results in generating only one or a few identical images repeatedly.

Identifying a mode collapse can be done by examining a large sample of generated images. These images will display limited diversity, with the same or a small subset of identical images appearing multiple times. Another way to detect a mode collapse is by reviewing the line plot of the model's loss over time. The generator model, in particular, will exhibit oscillations in its loss as it gets updated, jumping between different modes that yield varying loss values.

Convergence Failure

One of the most common challenges encountered during GAN training is the convergence failure [4]. Typically, this occurs when the neural network's model loss fails to stabilize during the training process. In the context of GANs, a failure to converge implies the inability to achieve a balanced equilibrium between the discriminator and the generator. Identifying this type of

failure often involves observing the discriminator's loss, which may decrease to zero or near-zero values. In some instances, the generator's loss may also rise and continue to increase over the same period.

This specific loss pattern is commonly caused by the generator producing poor-quality images that the discriminator can easily discern as fake. Such a failure mode might manifest early in the training and persist throughout the process, warranting an interruption in the training procedure. However, certain unstable GANs might experience this failure mode for several batch updates or even multiple epochs before eventually recovering.

3.2.5 Evaluation

Comparing one model to another is a common in machine-learning research, especially to showcase the superiority of a newly developed model in capturing specific distributions compared to existing models. However, this task can be quite difficult.

Unlike other deep learning neural network models that are trained using a loss function until convergence, GAN generator models rely on a second model called a discriminator, which learns to classify images as real or generated. Both the generator and discriminator models are trained together to achieve an equilibrium. Consequently, there is no direct objective loss function used to train GAN generator models, making it challenging to assess training progress or the absolute quality of the model based on loss alone.

Instead, a combination of qualitative and quantitative techniques has been devised to evaluate the performance of GAN models based on the quality and diversity of the generated synthetic images. Since there is no universally agreed-upon method for evaluating a given GAN generator model, assessment is often conducted by investigating the quality of the generated

images, typically within the context of the target problem domain.

Manual Evaluation

The visual inspection of samples by humans is a common and intuitive approach to evaluate GANs [3, 4]. Many GAN practitioners resort to manually assessing the images synthesized by a generator model as a means of evaluating GAN generators. This process involves utilizing the generator model to produce a batch of synthetic images and then evaluating the quality and diversity of these images with respect to the target domain. Such evaluations may be conducted by the researchers or practitioners themselves.

During GAN training, the generator model undergoes iterative updates over numerous training epochs. As there is no objective metric for model performance, determining when to stop the training process and when to save a final model for later use becomes challenging. To address this, it is common practice to utilize the current state of the model during training to generate a large number of synthetic images and save the corresponding generator model's state. This enables subsequent evaluation of each saved generator model through the examination of the generated images. A training epoch represents one complete cycle through the images in the training dataset, which is used to update the model.

While manual inspection serves as the simplest method of model evaluation, it comes with several limitations, including:

- Included biases due to the subjectivity of the reviewer about the model, its configuration, and the project objective.
- Requirement of knowledge of what is real and what is synthetic for the target domain.

- Time limitation. It is very time consuming and energy draining to go through many images and evaluate them as objectively as possible.

The inherent subjectivity in visual evaluation almost certainly results in biased model selection and cherry-picking, making it unsuitable for final model selection in non-trivial projects. However, it can serve as a useful starting point for practitioners to become aware with the technique. Thankfully, more advanced and sophisticated GAN generator evaluation methods have been proposed and embraced to address these limitations.

Qualitative Evaluation

Qualitative evaluation involves assessment of the quality of the synthetic data by domain experts by comparing them to real-world examples. Ideally, this is performed as a blinded experiment, where the experts are asked to predict if a certain datapoint is real or was generated and optionally provide feedback.

Some other qualitative measures include [3]:

- Nearest Neighbors - generated samples are displayed next to their nearest neighbors in the training dataset in order to detect overfitting
- Rapid Scene Categorization - participants must evaluate samples as real or fake in very short timespan
- Investigating and Visualizing the Internals of Networks

Quantitative Evaluation

Quantitative evaluation involves calculation of a specific score reflecting the current model state. Several such metrics have been proposed [3], for example:

- Average Log likelihood measures how well the distribution of generated images explain each real image. The assumption is, that each real image comes from a certain probability distribution (e.g. Gaussian).

- Inception Score (IS) uses a pre-trained neural network (such as a network trained on the ImageNet dataset) to classify data into one of its classes, with higher Inception Scores indicating that the generated images are of better quality and are more diverse in terms of different classes they represent.

- **Frechet Inception Distance (FID)** The FID score measures how similar real world data is to the generated data. A multivariate gaussian is fitted both to real and generated samples, where the generated samples are represented by a neural network embedding. FID is then the Wasserstein-2 distance between these two distributions, with low scores representing small difference between real and generated samples, i.e. [3]:

$$FID(r, g) = ||\mu_r - \mu_g||_2^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}), \tag{3.4}$$

  where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance matrix of real and generated distributions.

3.2.6 Types of Architectures

Depending on the application, there are variations of GAN architectures, which can be used. New uses of GANs as well as improved GAN techniques are continuously being researched and we mention some of the possible GAN variations:

- Progressive GANs (PGGANs) [13] represent a distinctive adaptation of GANs tailored for generating high-resolution images. Unlike traditional GANs, which operate solely at a fixed resolution, PGGANs initiate their training with low-resolution images and progressively advance to higher resolutions as training proceeds. This unique approach enables the network to efficiently learn the intricate details of high-resolution images, ultimately resulting in the production of realistic images.

- A Conditional Generative Adversarial Network (cGAN) [22] is an advanced version of the Generative Adversarial Network that can generate new samples based on specific conditions. Unlike a regular GAN that relies solely on random noise, a cGAN's generator takes both random noise and a condition, like a label, image, or text, to produce a sample specifically tailored to that condition. The discriminator in a cGAN also considers the condition when determining whether a sample is real or generated and matches the given condition.

- StyleGAN [16] is a generative model introduced by NVIDIA researchers. It can create exceptionally realistic images of faces and various objects. The model utilizes a technique called style transfer, enabling it to manipulate specific aspects of an image, such as pose, lighting, and expression while maintaining a high level of realism.

Chapter 4

Related Work

Nowadays, deep learning techniques are widely used for solving many complex tasks including image synthesis. Generative models have evolved rapidly over the past few decades and made significant advancements in their results in recent years. We describe in this chapter more in detail some of the architectures specifically designed for generating synthetic images.

4.1 Generating Synthetic Images

There are many types of models and architectures that can be trained to generate synthetic data, especially images. Generating high-quality, diverse and realistic images has been always a challenging task and therefore researchers are constantly developing new architectures to improve the quality of images and stabilize the training process. In the past, there were traditional techniques such as texture synthesis [5] or texture mapping [9] that could be used for image generation, but had some limitations. These methods were unable to create complex and diverse images and were directly dependent on hand-designed features of the algorithms. With the advent of deep learning

new ways of creating synthetic data were invented and described which made major progress in this area. One of the models that belongs into the category of deep generative architectures for image synthesis is a Variational Autoencoder, which we have already mentioned in Chapter 3. Year 2014 was a significant milestone for the field of generative modeling and image synthesis because Generative Adversarial Network was first proposed by Goodfellow et al. [7]. This sophisticated approach for generating images, where a generative model and a discriminative model play a minimax two-player game, has demonstrated impressive results and many extensions of GANs have been already developed including Progressive Growing GANs and StyleGANs which we further introduce.

4.2 Progressive Growing Generative Adversarial Network

A Progressive Growing Generative Adversarial Network (PGGAN) [13] was developed as an extension of the GAN architecture in 2017 by NVIDIA researchers with an aim to make the training process more efficient and improve the quality of generated data. The architecture is based on incrementally increasing resolution of images during the training which remarkably stabilizes the learning process and enables synthesis of high-resolution realistic images at the end. Producing high-resolution images has been a challenging task for GANs before because a generator model had to learn the overall structure of an image as well as fine details at the same time. PGGAN solves this problem by adding new convolutional layers to both generator and discriminator models throughout the training. All images of the training dataset need to be downsampled to lower resolutions before the training process because the

networks take images in various resolutions as input during learning. Adding new layers to both networks throughout the training enables to increase the resolution which is visualized in Figure 4.1. The training starts with low resolution where generator produces images of size e.g. 4x4 pixels and discriminator classifies low-resolution images as real or fake. When the training converges to some extent, new layers are faded in to both networks and the training continues with higher resolution (usually the number of the pixels is doubled from the previous iteration of a model architecture, so the next resolutions are 8x8, 16x16, etc.) and this process is repeated multiple times until the desired resulting resolution is reached. This training methodology greatly stabilizes the training, because going straight from latent space to high resolution contained vast amount of variance in the previous GAN research and therefore it was difficult and almost impossible to generate realistic high-resolution images.

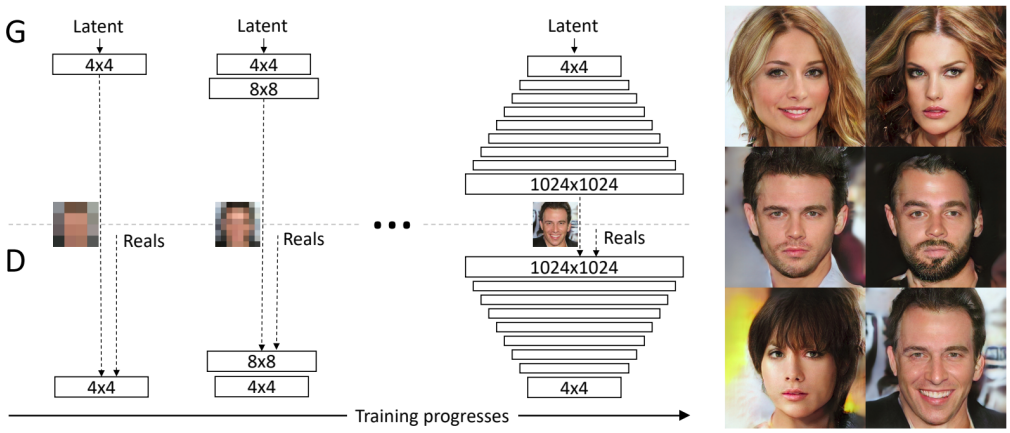


Figure 4.1: Schema of the multi-scale architecture used in PGGAN and six examples of generated synthetic images with resolution 1024x1024 pixels [13].

The generator and discriminator networks are mirror images of each other and grow at the same pace throughout the training. Adding systematically

new blocks of convolutional layers enables the generator and discriminator models to learn and focus on high-level structure at first and later on learn finer details as the training progresses.

Further, we describe the way how new blocks of layers are added to both networks which is also illustrated in Figure 4.2. New layers are smoothly faded in to the existing architecture rather than adding them directly, since it avoids destabilization of the already well-trained model suitable for a smaller resolution. Additionally, all layers including existing layers as well as new added layers remain trainable during the whole learning process. The training process involves periods of a model fine-tuning and periods of growing the resolution and also networks via slowly phasing in new layers. Inserting new block of layers includes creating new skip connections between the new block and the existing input of the discriminator or existing output of the generator. These connections are weighted which regulates the impact of the new added block on the PGGAN model. The weight depends on the hyperparameter *alpha* which is equal to zero or any small number close to zero when new layers are inserted and increases linearly up to 1 during the gradual incorporation of the new block. We further illustrate the changes of the discriminator and the generator models during the process of phasing in the larger resolution from 16x16 to 32x32 pixels which is also visualised in Figure 4.2. The discriminator model needs to grow and insert new input layer to support the larger input image size (32x32 pixels) followed by the inclusion of a new block of layers. The output of an inserted block is afterwards downsampled to the previous resolution (16x16 pixels). Simultaneously, the new image itself is downscaled directly to the lower resolution (16x16) so it can be directly passed through the original layers of the discriminator. The downsampling of images is done using average pooling. The transition to

higher resolution is summarized in the following two pathways:

- 32×32 Image → fromRGB Convolution[2] → New Block → Downsample → 16x16 Image1
- 32×32 Image → Downsample → fromRGB Convolution[2] → 16x16 Image2

Both 16x16 output images from these pathways are subsequently combined by taking their weighted average, where the weighting is controlled by the hyperparameter *alpha*, as follows:

$$output = (1 - alpha) * Image2 + alpha * Image1. \tag{4.1}$$

The subsequent step involves feeding the weighted sum of outputs from the two pathways into the rest of the existing model. During the initial stages, the weighting completely favors the output of the old input processing layer ($alpha = 0$). As the hyperparameter *alpha* is linearly increasing through the training iterations, the new block gains more weight over time until it becomes the only contributor to the output ($alpha = 1$) and the old pathway is entirely eliminated.

The growing process of the generator model is similar but inverse as we described above for the discriminator. Initially, the output of the last block with lower resolution (16x16 pixels) is upsampled to higher resolution (32x32 pixels) using a nearest neighbor method. This upsampled output is then connected to the new block which is designed for the increased image size. During the phase-in period, the upsampled output is additionally linked to the output layer of the previous model. These two output pathways of

[2]Conversion of RGB colors to a feature vector implemented as a 1x1 convolutional layer

the generator model during the transition phase are following (from 16x16 to 32x32 pixels):

- 16×16 Image → Upsample → New Block → toRGB Convolution[3] → 32x32 Image1
- 16×16 Image → Upsample → toRGB Convolution[3] → 32x32 Image2

Both outputs are merged into one output using weighted sum as before at the discriminator model, allowing for a smooth transition. Once the phase-in process is accomplished, the old output layer becomes redundant and is removed.

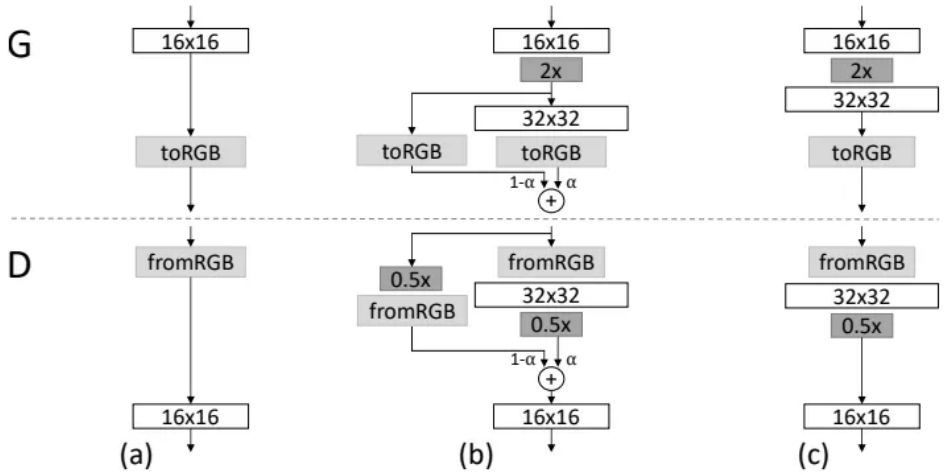


Figure 4.2: Diagram showing a transition of the model from the resolution of 16x16 to 32x32 pixel color images (a) before growing, (b) during the fading in of the larger resolution, and (c) after the transition phase. The 2x and 0.5x blocks represent upsampling and downsampling of images respectively. The toRGB and fromRGB layers represent projections between a feature vector and an RGB color image using convolution of size 1x1.

Overall, the training methodology of PGGAN offers several advantages

[3]Conversion of a feature vector to color image using convolutional layer consisting of 3 filters of size 1x1

over traditional GAN architectures including generation of high-quality images, stable training and hierarchical feature learning. This approach also makes the training process more effiecient and reduces training time up to $2 - 6$ times since most of the iterations run at lower resolutions. The exact time savings depend on the chosen final resolution of images. However, achieving true photorealism still remains challenging and there is room for improvement in the fine details of the synthetic images generated by PGGAN.

4.3 Style Generative Adversarial Network

One of the renowned models for image synthesis, which was introduced by NVIDIA in 2018, is Style Generative Adversarial Network (StyleGAN) [16]. The GAN architecture, especially the generator part of the model, is considered as a black box in generating data, because it is not possible to control and regulate different aspects of the output image e.g. pose, face shape, and hair color. StyleGAN architecture proposes some changes how to control the synthesis procedure and the characteristic of the generated image. This type of architecture is based on the progressive GAN, uses its training methodology and changes only the generator model of the PGGAN. We further describe the modifications which were introduced by the authors and involved into the architecture. The architecture of the generator is illustrated in Figure 4.3 and consists of the synthesis network and the mapping network. Instead of feeding the generator, specifically the synthesis network, directly with a latent code the StyleGAN takes a tensor of size 4x4x512 as an input which is learned during the training. The architecture of the generator is extended by the mapping network which contains 8 fully connected layers.

This multilayer perceptron takes a latent code z as an input and produces vector w as an output which is also called a style vector. The dimension of both vectors z and w is set to 512. The style vector is afterwards transformed via affine transformation which is implemented using two layers (block A in Figure 4.3) and provides the style $y = (y_s, y_b)$ where y_s represents scale and y_b bias respectively. The style is then incorporated through adaptive instance normalization (AdaIN) operation into the each block of the synthesis network. The AdaIN equation looks as follows:

$$AdaIN(x_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i}, \tag{4.2}$$

where the feature map of the previous convolutional layer x is normalized to obtain the Gaussian distribution with zero mean and unit variance. Then the style with scale y_s and bias y_b is applied and influences the impact of feature map channels on the subsequent convolution layer.

Next, a Gaussian noise in form of a single-channel image is incorporated into each layer of the synthesis network before the AdaIN method. Block B, displayed in Figure 4.3, applies additionally learned per feature scaling factors to the noise and the output images are added to the activation maps after the convolution. The injection of noise enables to generate stochastic variation at each level of detail. The stochastic variation in an image refers to minor details which do not significantly alter the overall context of the image such as placement of hair, freckles and smile angle.

The next difference, which was introduced by authors of StyleGAN in comparison with the PGGAN, is replacement of the nearest-neighbor upsampling method with bilinear sampling. This approach involves applying a second-order binomial filter to the activations after each upsampling layer and before each downsampling layer which improves the result image quality.

Overall, the StyleGAN architecture solves the entangled problem of the previous GAN and also PGGAN architectures which means that one element of the latent code influences more than one feauture of an image which is undesirable. By introducing the mapping network the intermediate latent space W becomes disentangled and enables to control each characteristic of an image separately. It is also possible to scale one aspect of an image in StyleGAN without affecting the others e.g. changing the hair length.

The authors of this architecture also demonstrate that the redesign of the generator even enhances the output quality of images with the advantage of controlling the visual attributes of generated images.

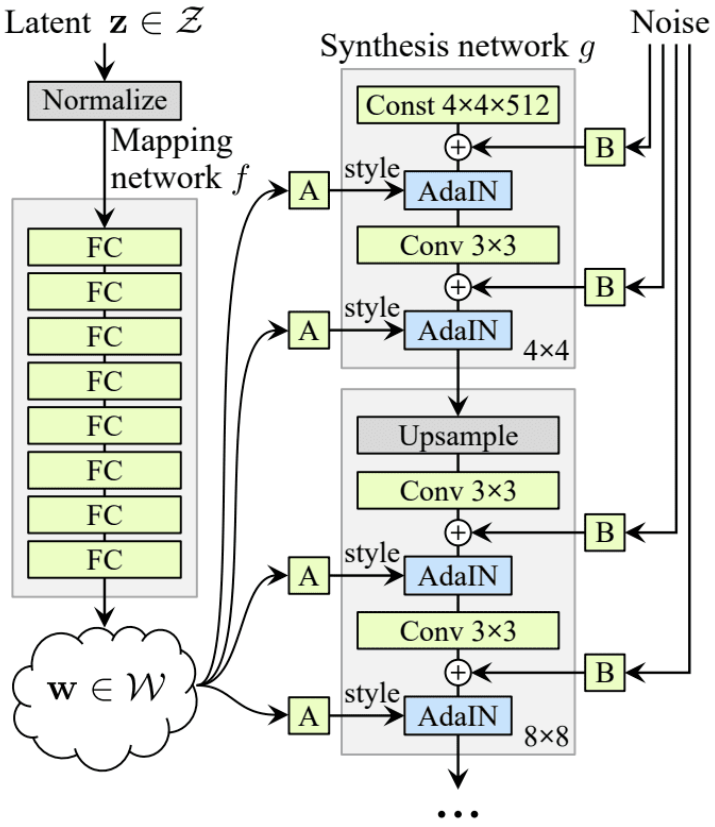


Figure 4.3: The architecture of style-based generator [16].

4.4 Generative Adversarial Transformer

The Generative Adversarial Transformer (GANformer) [11], which was introduced by Hudson et al. in 2021, has brought a new insight into the field of generative models and combines the GAN architecture with Transformer. The Transformer [26] is a neural network based on self-attention mechanism which means that it can process any part of the input sequential data, capture long-range interactions and understand complex patterns within data. Primarily, the Transformer is used for natural language processing tasks, but after incorporating the Transformer into the GAN model, we obtain the GANformer, which is specifically designed for generative modeling of images.

The GANformer has a bipartite structure which means that there are interactions between two sets of variables - the latents and the image features. Two novel forms of attention displayed in Figure 4.4 are introduced by the authors and can be used in GANformer - simplex or duplex. The simplex attention propagates information only in one way from latents to visual features and the duplex attention distributes information in both directions.

Design of the GANformer architecture is similar to StyleGAN. It consist of the discriminator and the generator and also follows the training methodology and the configurations of the StyleGAN model. The key difference lies in incorporation of the bipartite attention layer which enables to perform spatial decomposition in comparison with StyleGAN where style of all features is controlled rather globally. The latent vector z is splitted into k components (k is from the range of $8-32$), $z = [z_1, z_2, ..., z_k]$ which is shown in Figure 4.5. These latent variables are further transformed by a shared mapping network into a set of k intermediate latent variables $Y = [y_1, y_2, ..., y_k]$. During the image synthesis, the feature map X and latent variables Y take on the roles of two element groups, exchanging information through the attention layer,

which is achieved via either the simplex or duplex attention operation.

The training process starts with a 4x4 grid, which is fed into multiple layers including the attention layer for the exchange and aggregation of information about semantic regions in an image, the convolution layer and the upsampling operation. This block of layers is repeated until reaching the resulting higher resolution (e.g. 256x256 pixels).

The discriminator which predicts whether image is real or fake has the inverse model structure as the generator. It comprises the convolution layers and downsampling layers which reduce gradually the resolution until the discriminator makes a prediction. The attention layer can be optionally inserted into the discriminator model to collect the information from the processed image, but the authors did not observe visible improvement in model performance when applying the bipartite attention.

To summarize, the bipartite attention used in the GANformer has brought an advantage of controlling the style of whole semantic regions which enables the generation of structured multi-object scenes in high quality while maintaining linear computation efficiency.

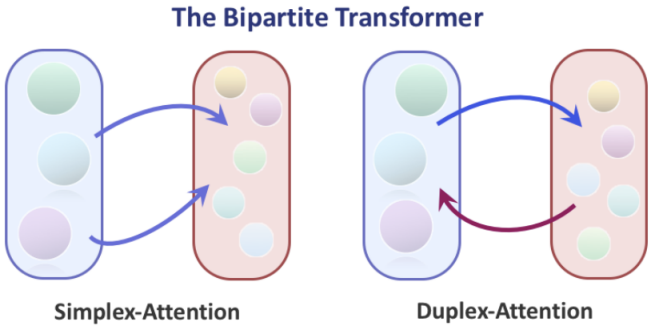


Figure 4.4: Two novel forms of Bipartite Attention depending on the direction in which information propagates - simplex attention (one way only) and duplex attention (both ways) [11].

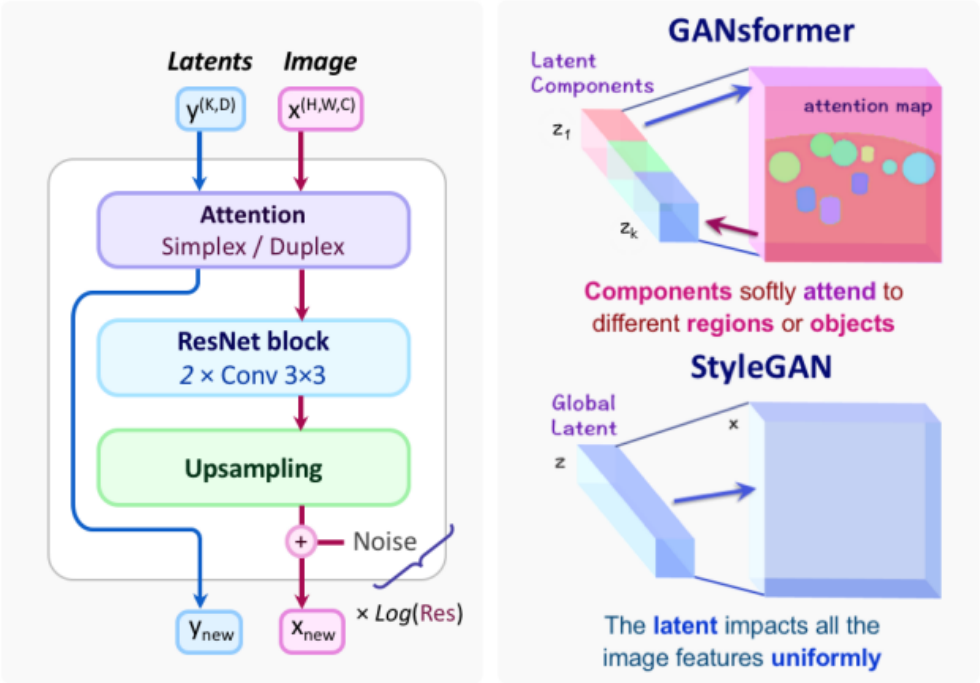


Figure 4.5: Architecture overview. Left: The GANformer consists of the blocks of layers comprising a bipartite attention (simplex or duplex), followed by convolution and upsampling. Simultaneously the latent variables Y are updated to capture semantic information of the synthesized image. Right: The latents and image features interact with each other to capture the scene structure. In contrast to StyleGAN where a single latent modulates the entire image uniformly, GANformer adopts a compositional latent space, allowing for more localized control over different semantic regions of the generated output image. [11]

Chapter 5

Research

In this chapter, we describe the data preprocessing as well as methods we decided to use for the image synthesis. We introduce and explain four different architectures we utilized for the image generation and provide insights into the training process of each model.

5.1 Data Preprocessing

All data was downloaded using the written Python script and needed to be further manually controlled. Dentists who submitted treatments of their patients made in some cases a mistake and as a result some photos were wrongly assigned to a view what we corrected. There were also occasionally some missing photos in treatments and therefore there were slightly different amounts of photos for each view.

The most problematic part about the dataset was the variety of resolutions, which had to be unified to make the images usable for model training. We set the resolution to 256x256 pixels and created two datasets: one with padded versions of images and another with center-cropped images (see

Figure 5.1). The models were configured to generate images in the same resolution of 256x256 pixels.

The manual inspection and filtration of the photos was extremely time consuming but necessary to have the consistent training dataset of images. There were some blurred images and images containing any label, mostly a company logo, which were excluded from further research. If the original photo contained some padding or was photographed from a long distance we cropped such an image. Next, we removed all images with the aligners put on the teeth since there were only several images like that and it would unnecessary increase the variation in the dataset.

We also filtered out all images which had a width lower than 256 pixels since we decided to generate images in resolution 256x256 pixels and did not want to have padding all around the image in the padded version of the dataset, but only at the bottom of an image.

Additionally, we flipped each left view image of teeth horizontally since the right and left views were the same symmetrically and created a merged side view dataset containing right and flipped left views. This effectively doubled the dataset size of this view. We also had to flip several images of upper and lower views vertically so the dental arch was always in the same position.

Overall, after the data filtration process the dataset consisted of 19,738 intraoral photos and the specific count for each view is displayed in Table 5.1.

View	Number of photos
Front	4,065
Right	3,818
Left	3,817
Upper	4,092
Lower	3,946

Table 5.1: Number of photos by view in the dataset.

(a) Original photo (b) Center cropped image (c) Padded image

Figure 5.1: Real images of the front view in original, cropped and padded version.

5.2 Architectures and Training

We researched many architecture types for image synthesis which are popular nowadays in the machine learning community and trained some of them which is summarised more in detail in this section. All architectures were trained on remote machines with 2 GPUs each with 12GB of memory.

Regarding the training dataset, we decided to use the front view of teeth as the default and determining view for the comparisons between architectures, because these images looked the least variable in terms of camera angle as we inspected the whole dataset. To achieve the best possible results from each model, we opted for the front view with low diversity since our dataset for each view did not contain as many images as the state-of-the-art methods used.

There is no one-size-fits-all rule for determining when to stop the training of a generative model. Early stopping technique can be used when the quantitative metric does not improve, the training process can be stopped when the metric reaches a satisfactory level or the fixed number of epochs can be defined. It is also possible to manually inspect the generated images visually and observe whether the quality of synthetic images is improving or rather starts to degrade.

5.2.1 Deep Convolutional Generative Adversarial Network

In the initial stages of developing this thesis, we decided to train a basic Deep Convolutional Generative Adversarial Network (DCGAN) [23], which unfortunately did not produce sufficient results which could be usable in some real-world applications. This concrete type of network implementation[4] was too simple and has already become quite outdated because of achieving only lower quality results. We modified the network by adding some layers to increase the resolution from original 64x64 to 128x128 pixels. One example of generated image is shown in Figure 5.2, where it can be directly seen that this image is synthetic and has too low quality.

5.2.2 Progressive Growing Generative Adversarial Network

We continued in the research and studied newer architectures suitable for the image synthesis which are capable of achieving better quality results. As the next one, we chose the PGGAN model [13] which offers improvements in the

[4]https://keras.io/examples/generative/dcgan_overriding_train_step

Figure 5.2: Synthetic image generated by the DCGAN model in resolution 128x128.

training process in terms of time reduction and higher stability of the model during the learning as well as ability to produce images of higher resolutions. The architectures of the generator and discriminator consist mainly of 3-layer blocks which are gradually added to both networks during the training as we have already described in Chapter 4. Each block consists of 2 convolution layers of size 3x3 and an upsample or a downsample part for a generator or discriminator network respectively.

As the optimizer we used Adam [17] with the learning rate $\alpha = 0.003$, $\beta_1 = 0$, $\beta_2 = 0.99$ and $\epsilon = 10^{-8}$ and we used leaky ReLU as an activation function after each convolution operation. The latent space size was set to 512 components, which were randomly sampled from the standard normal distribution with $mean = 0$ and $variance = 1$ and representend input latent vector for the generator network.

We started with the resolution 4x4 pixels which was doubled throughout the training process. The growing of both networks continued with each block having twice resolution of the previous one until the desired resolution was reached. As we have already mentioned we trained all networks using

the images of size 256x256 pixels, so we set the goal to generate images in the same resolution of 256x256 pixels. Therefore, we had to leave out 2 blocks from both ends of the networks since the default output resolution was set to 1024x1024. The number of epochs for each stage was set to value of 42.

Overall, we trained the PGGAN twice for the images displaying the front view of teeth with the same network settings using two different training datasets. The first dataset consisted of images with the black padding and the other one contained cropped images. As the training loss we chose the Wasserstein Gradient Penalty Loss (WGAN-GP) [8] which improves the training stability. The equation of WGAN-GP loss is as follows:

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] + \lambda\ \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2]. \qquad (5.1)$$

Verbally described the loss consists of two parts - Wasserstein loss [2] on the left side of the sum and the penalty of the gradient norm which is on the right of the sum. The first part is calculated as the difference between the average output of the discriminator on real samples and on generated samples which represents the distance between the real distribution and the model distribution of the data. The second part is the squared difference of the gradient norm and 1 where $\mathbb{P}_{\hat{x}}$ is the distribution achieved by uniformly sampling along a straight line between real and generated pairs of samples. The gradient norm is therefore encouraged to be equal to one to minimize the penalty term. Additionally, there is a coefficient λ which weights the gradient penalty term and was set to 10 during the training.

5.2.3 Style Generative Adversarial Network

The renowned GAN architecture which we have already mentioned and introduced in Chapter 4 is called StyleGAN [16]. We used the newest official implementation[5] of this model from authors Karras et al. at NVIDIA [15]. We followed the recommended configurations by authors for the selection of appropriate training options. We utilized the StyleGAN3-R network implementation which is translation and rotation equivariant. The model was trained using non-saturating logistic loss [7]:

$$f(x) = log(sigmoid(x)), \tag{5.2}$$

with R1 regularization [21], setting the regularization weight γ to 2. We also utilized the Adam optimizer [17] with $\beta_1 = 0$, $\beta_2 = 0.99$ and $\epsilon = 10^{-8}$ during the training to achieve better performance. For the augmentation mode we chose the adaptive discriminator augmentation [14] in order to avoid overfitting of the discriminator since we had only limited data and small training datasets.

The progressive growing of the StyleGAN network is identical to the implementation of PGGAN [13] but it starts with the resolution of 8x8 pixels instead of 4x4. The resulting image size was set to 256x256 which is the same as the resolution of the training data. Altogether, the generator model learned 18,816,387 parameters and the discriminator 28,864,129 parameters.

At first, we trained the model for the front view with and without padding which has shown promising results especially for the cropped images. Therefore, we decided to utilize the StyleGAN model using cropped images of each view of teeth as well as the mixture containing data from all views. Since all

[5] https://github.com/NVlabs/stylegan3

views except the side view are horizontally symmetric, we enabled the data augmentation with random x-flips for the front, upper, lower and mixed views during the training process which efficiently doubled the number of images. We disabled the mirror augmentation for the side view of teeth.

The training loop exported every 80 iterations the network snapshots, generated a grid with randomly sampled fake images, and evaluated selected metrics, which was the Frechet Inception Distance (FID) in our case. All these results were logged for further analysis and monitoring.

5.2.4 Generative Adversarial Transformer

The GANformer [11] which unifies the GAN and the Transformer architectures is one of the newest generative models for the image synthesis. We trained the newest implementation[6] of this model released by the authors of GANformer model using the Pytorch version. Most of the training configurations and settings were adopted from the StyleGAN2 model. As the loss function we used a non-saturating logistic loss with R1 regularization. For optimization, we utilized the Adam optimizer [17] with a batch size of 32. We set the hyperparameters of the Adam optimizer as follows: learning rate $\alpha = 0.002$, $\beta_1 = 0$ and $\beta_2 = 0.99$. Leaky ReLU activations with $\alpha = 0.2$ were employed after each convolution along with bilinear filtering in all upsampling and downsampling layers. The latent size was set to the default value of 512 variables. The mapping part of the generator model consisted of 8 layers and ResNet connections were incorporated throughout the model including the mapping network, synthesis network, and discriminator. Overall, the number of parameters for the generator model was 30,903,632 and 28,864,129 for the discriminator model respectively. We trained the GANformer model on

[6] https://github.com/dorarad/gansformer

the dataset of the front teeth, both with and without padding. The mirror augmentation was applied to the images to increase the effective training dataset size.

Chapter 6

Results and Discussion

In this chapter, we provide and discuss the results of all trained models. For the evaluation, we used the quantitative metrics as well as the qualitative assessment from dental experts.

6.1 Quantitative Evaluation

There are different types of metrics, which we have already mentioned in Chapter 3, suitable for the evaluation of the quantitative performance of a generative model. We decided to use the FID score [10] since this metric is commonly employed for evaluating generative models in scientific papers [10, 11, 13, 15, 16]. The FID score serves as a measure of similarity between two sets of images, with lower scores indicating greater similarity or statistical likeness. A perfect FID score of 0 signifies complete identity between the two groups of images. Empirical evidence has demonstrated that lower FID scores correspond to higher quality images [10].

6.1.1 Progressive Growing Generative Adversarial Network

In Figure 6.1 we display several examples of the generated images produced by the PGGAN model and the FID scores are shown gradually for each depth of the generator model during the training in Figure 6.2. The metric was computed for each model depth representing specific resolution, ranging from 4x4 to 256x256 pixels, every 10 epochs throughout the training process. The lowest FID score value at the 256x256 resolution was 331.59 for padded images and 306.79 for cropped images respectively. The observed image quality appears far from realistic and it is evident that these images are synthetic. Nevertheless, the model managed to learn the characteristics of the images to some extent. However, the obtained results are not suitable for real-world applications.

6.1.2 Style Generative Adversarial Network

We trained the StyleGAN model using 6 different datasets, which included the front view in both padded and cropped versions, cropped images for the side, upper, and lower views, as well as the mixture of all views. Authors claim that it is possible to achieve reasonable results after 5,000 training iterations [15], so we set the number of iterations slightly higher to 5,200 steps. We trained the model on cropped images of the front view for 5,200 iterations, but for the padded images we stopped the training earlier after 1,600 steps since the model showed much better performance on the cropped images. Additionally, the training process of the model learned on padded images had stability issues, which were reflected in the computed FID score (see Figure 6.3). Therefore, we trained the rest of the views using only cropped

Figure 6.1: Synthetic images with and without padding generated by the PGGAN model in resolution 256x256 pixels.

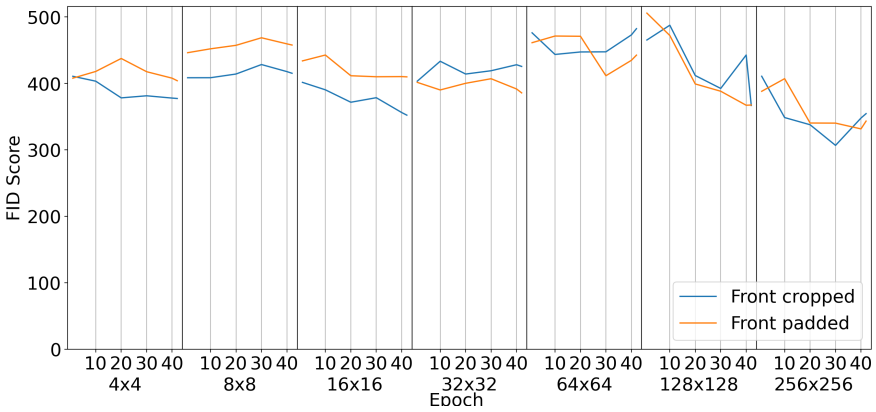


Figure 6.2: PGGAN training progression. On the y axis is the FID score. The x-axis contains number of epochs for each of the resolutions.

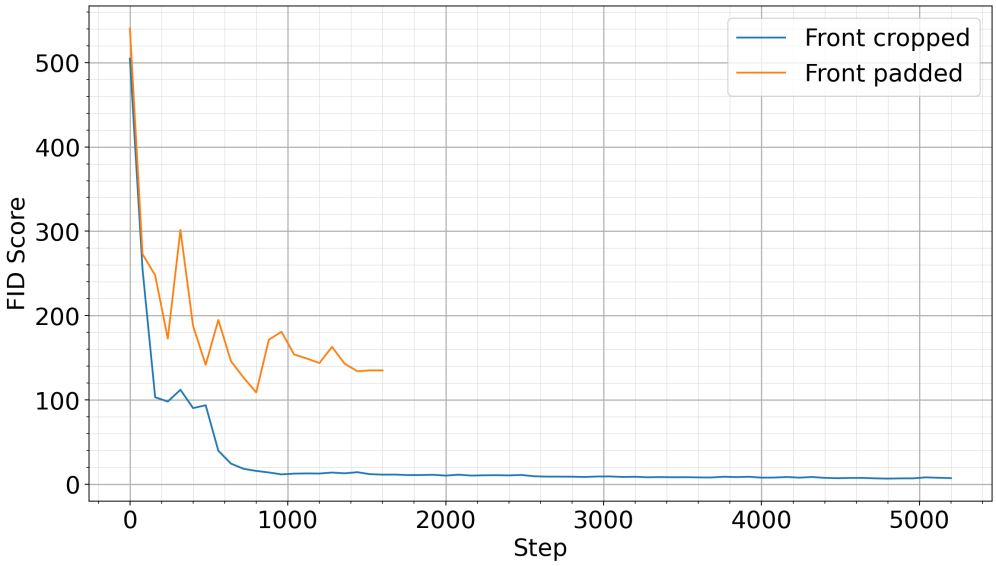


Figure 6.3: StyleGAN training progression on front teeth view with comparison between padded and cropped inputs.

images. However, the default setting of the training duration was 25,000 iterations, but due to our time limitations we stopped training earlier. We also noticed that the quality metrics started to oscillate around the 1,500th iteration and did not change further rapidly so with prolonged training we would probably not obtain significantly better results (see Figure 6.4). Since we expected similar convergence scenario of the model for the other views, we set the number of iterations to value of 3,600. The lowest FID score values for the front padded, front cropped, side, upper, lower, and mix datasets are shown in Table 6.1 and were achieved in iteration steps 800, 4800, 3440, 3440, 2640, and 3600 respectively. We took the network snapshots with the lowest achieved score for each trained StyleGAN model and generated several synthetic images. Three example fake images per view from each model are displayed in Figures 6.5, 6.6, 6.7, 6.8, 6.9, and 6.10.

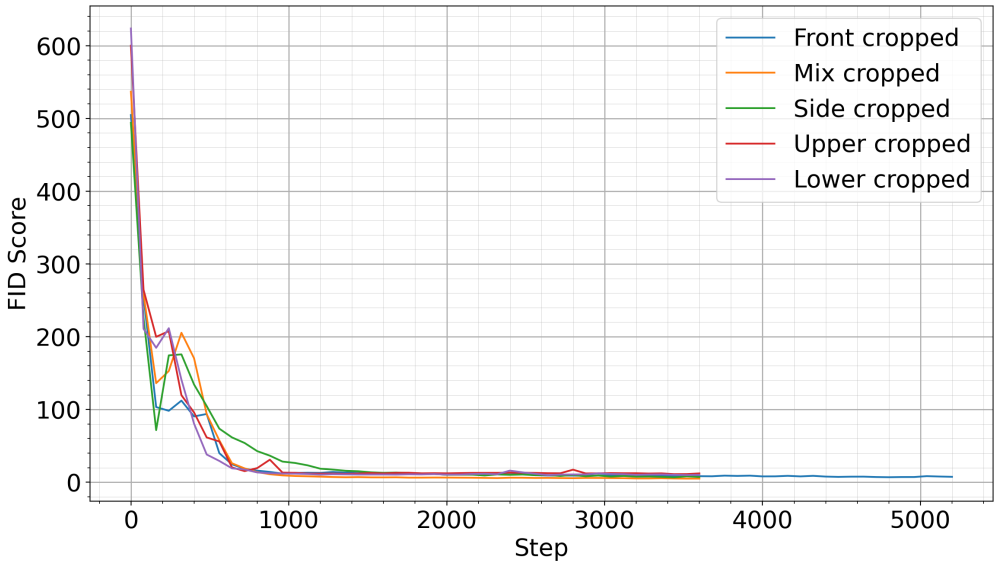


Figure 6.4: StyleGAN training progression on different teeth views only on cropped inputs.

Figure 6.5: Padded synthetic images of the front view generated by the StyleGAN model in resolution 256x256 pixels.

Figure 6.6: Cropped synthetic images of the front view generated by the StyleGAN model in resolution 256x256 pixels.

Figure 6.7: Cropped synthetic images of the upper view generated by the StyleGAN model in resolution 256x256 pixels.

Figure 6.8: Cropped synthetic images of the lower view generated by the StyleGAN model in resolution 256x256 pixels.

Figure 6.9: Cropped synthetic images of the side view generated by the StyleGAN model in resolution 256x256 pixels.

Figure 6.10: Synthetic images generated by the StyleGAN model using the dataset containing mixture of all views (front, upper, lower, left, and right) in resolution 256x256 pixels.

6.1.3 Generative Adversarial Transformer

The resulting FID scores, computed during the GANformer training process at every 24th iteration, are displayed in Figure 6.11. The model trained on padded images performed significantly worse compared to the model trained on cropped images. This trend has already been observed in the previous models we have mentioned before. We stopped the training on the images with padding at the 2112th step since the score increased rapidly. The training process using cropped images oscillated for many iterations, and because of that, we ended the training at the 2520th iteration since we did not expect any further progress. The lowest FID score had a value of 65.13 for the padded version of images, while the cropped images achieved a score of 22.02. We took the networks with the lowest FID and generated several images which are shown in Figure 6.12. Overall, this model achieved inferior performance than the StyleGAN.

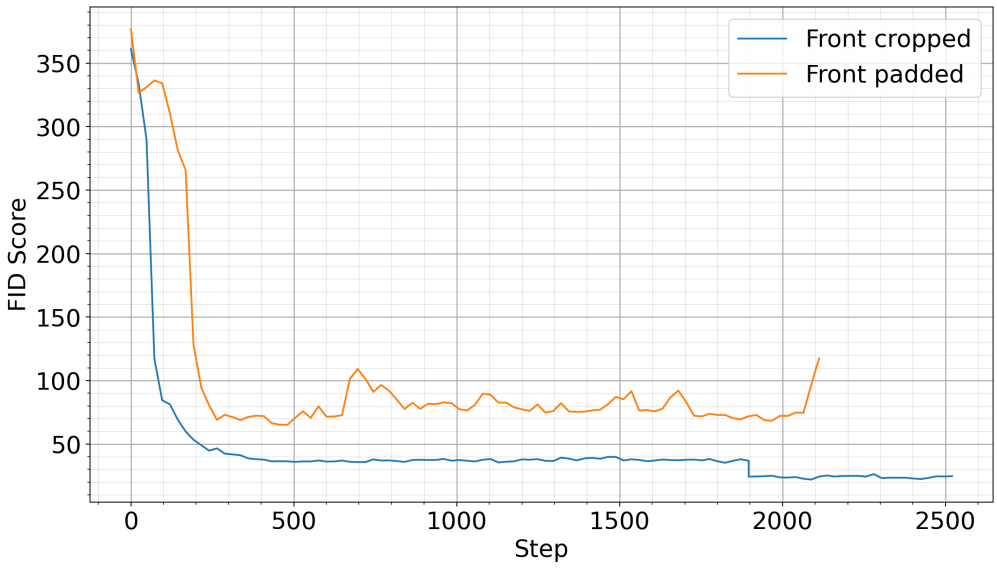


Figure 6.11: GANformer training progression on the front teeth view with comparison between padded and cropped inputs.

Figure 6.12: Synthetic images with and without padding generated by the GANformer model in resolution 256x256 pixels.

6.2 Qualitative Evaluation

We decided to perform the visual assessment of the generated intraoral images by dental experts. Initially, we manually inspected the generated images from the models with the lowest FID scores. Subsequently, we concluded that only synthetic images from the StyleGAN and GANformer models exhibited a level of realism worthy of qualitative evaluation by the experts (see Figure 6.13). This observation was further supported by the FID scores, with the PGGAN model yielding the highest values, indicating lower image quality.

We manually selected a total of 45 synthetic images without padding that seemed real for further qualitative evaluation. Additionally, we chose 25 real images specifically without dental anomalies so they were similar to the generated images and did not incorporate bias into qualitative results.

(a) Real (b) PGGAN (c) StyleGAN (d) GANformer

Figure 6.13: Comparison of real with synthetic images generated by PGGAN, StyleGAN and GANformer models.

Altogether the set comprised 25 real images (5 photos per view), 5 generated images from each StyleGAN model trained on one specific view (front, upper, lower and side), 20 images from the StyleGAN model trained on the mix of all views (front, upper, lower, left and right), and 5 images of the front view generated by the GANformer model. Afterwards, all images were randomly shuffled and we created a Google Form with selected images for the evaluation by the dental experts. The visual assessment of these images was performed by a total of 8 dentists, 4 dental hygienists and 3 dental assistants. 12 of them had less than 10 years of clinical experience, while the remaining 3 participants had over 12 years of clinical experience. The evaluators were not informed about the ratio of real and generated images and they were asked to determine whether each image was real or generated. To minimize the likelihood of random guessing due to time constraints, the evaluation time was set to be unlimited. This allowed each professional to assess the images carefully and without feeling rushed. At the end of the questionnaire, each participant was asked to pick one attribute or write their own characteristic that best differentiated between real and synthetic images. The predefined attributes, they could choose from, were teeth, alignment and soft tissue. The results of this question provided valuable insights into the aspects of the images that lacked the most realism and should be enhanced.

In Figure 6.14 we show comparison of each model trained on various datasets with the results from the Google Form. For each image in every dataset, we calculated the proportion of experts that selected that the image is real, which is shown on the y-axis ("Real" Selection rate).

It is clear, that the GANformer model performed much worse than the StyleGAN network and for almost all images all the experts clearly saw that the images were generated. However, StyleGAN seemed to perform

much better, on par with real images. When plotting all real images vs. all StyleGAN generated in Figure 6.15, with "Real" Selection rate on the y-axis, the mean "Real" Selection rate of all images is higher than that of the real images, meaning that the experts were more likely to select a StyleGAN generated image as real than the true real image. It is important to mention again that the samples were not chosen randomly from the predicted batch, but by manually inspecting the images and selecting representative ones most resembling true pictures of teeth. There were, of course, some synthetic images that lacked realism and were obviously fake on first glance.

Majority (11 out of 15) of the dental experts selected that the soft tissue was the tell-tale sign of the fakedness of the image. No one selected the teeth alignment, three chose the "teeth" option, and one dentist wrote own attribute, which included the center symmetry and shadows.

Type of model	**Training dataset**	**FID score**
PGGAN	Front padded	331.59
PGGAN	Front padded	306.79
StyleGAN	Front padded	108.96
StyleGAN	Front cropped	6.86
StyleGAN	Side cropped	6.7
StyleGAN	Upper cropped	11.12
StyleGAN	Lower cropped	9.84
StyleGAN	Mix cropped	4.97
GANformer	Front padded	65.13
GANformer	Front cropped	22.02

Table 6.1: Comparison of trained models displaying the lowest achieved FID score for each of them.

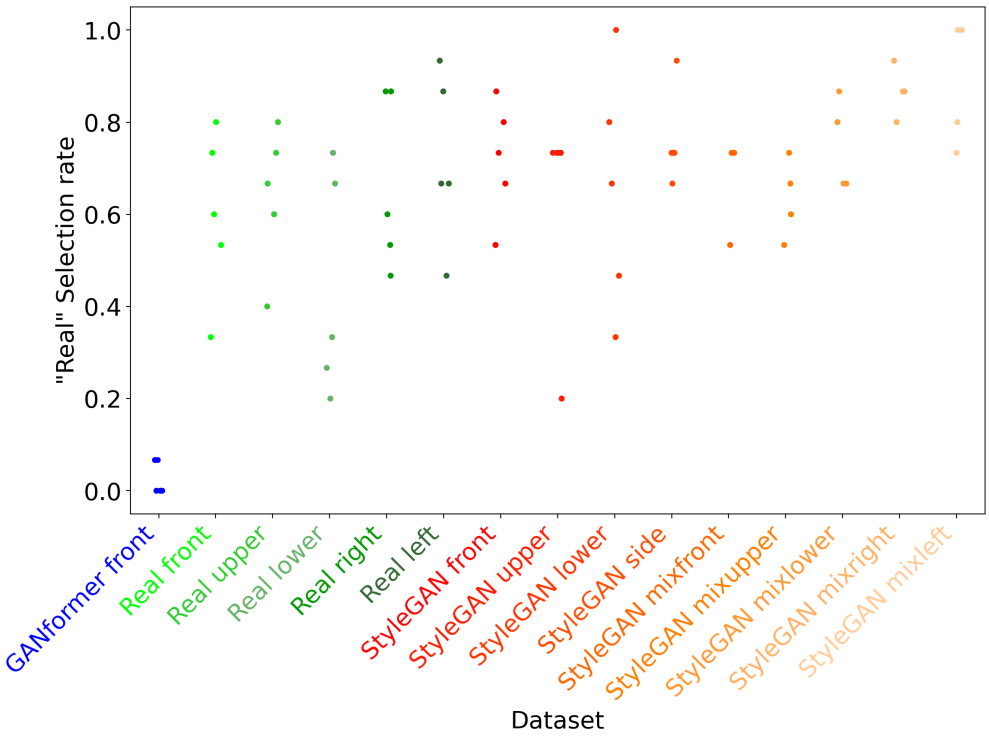


Figure 6.14: "Real" selection rate for each dataset in the evaluation. For each image, we calculated a mean of of predictions, where the choice "real" corresponds with number "1" and "generated" with number number "0". Each dataset thus contains as many datapoints as there were unique pictures in it, with each point representing how close to reality the experts found them to be.

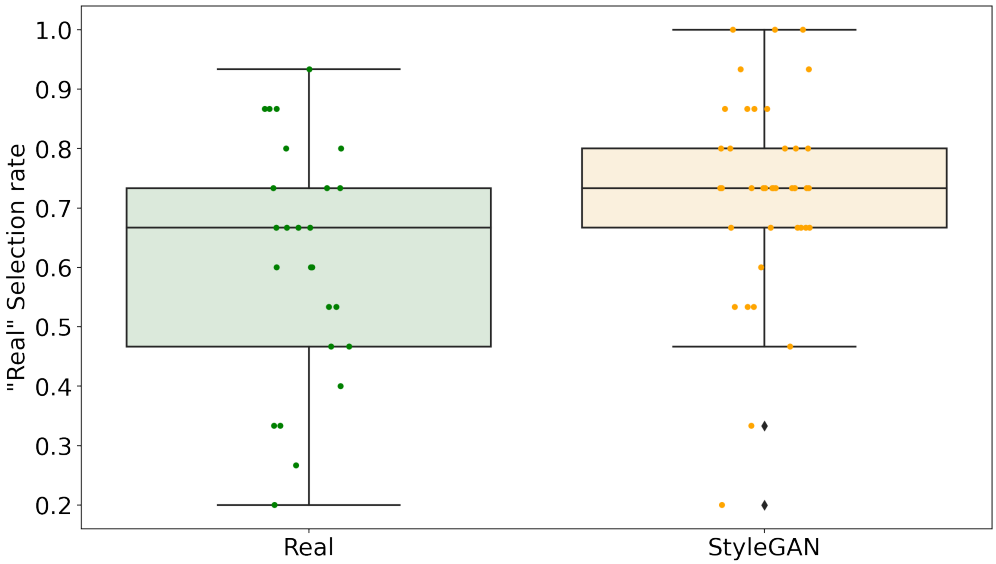


Figure 6.15: "Real" Selection rate of all truly real images versus all generated images by the StyleGAN network.

6.3 Comparison with Existing Methods

Authors Kokomoto et al. in their paper [18], published in 2021, utilized the PGGAN architecture to generate intraoral images. Their training dataset comprised 35,254 images of teeth and they trained the model with 4 different resolutions, concretely 128x128, 256x256, 512x512, and 1024x1024. The results were assessed quantitatively using the Sliced Wasserstein Distance metrics [13] as well as qualitatively which was performed by 12 pediatric dentists. Dental experts visually evaluated 50 real and 50 generated images in each mentioned resolution resulting in total of 400 images and classified each image as real or fake. Most of the dentists chose the teeth from the given four option of attributes (teeth, alignment, soft tissues, and others) as the assessment criteria when differentiating images. Kokomoto et al. claimed in conclusion that the resulting generated images of the resolution 512x512

or lower were considered to be indistinguishable by dentists, whether they were real or synthetic.

When comparing their results to ours using the PGGAN model, authors achieved slightly higher image quality. However, we trained the PGGAN model on the front view using only 4,065 images, which is almost 9 times less data in comparison with 35,254 samples. We also had very diverse images regarding the dental conditions, as these images were obtained from orthodontists and showed many dental anomalies. Nevertheless, we achieved much higher quality results using the newest architecture of the StyleGAN model [15], even with a smaller training dataset, in comparison with the PGGAN.

Conclusion

Generative models are gaining popularity as synthetic data can be widely used in various real-world applications. Especially in medical research, the lack of data and privacy restrictions can significantly slow down progress. Generating synthetic data can be invaluable for data augmentation of existing datasets, educational materials for students, as well as innovative applications to help doctors in diagnosing patients.

In our thesis, we worked with data obtained from the dentists in form of intraoral photographs made before, during and after the orthodontic treatments. We manually inspected the whole dataset, filtered out low-quality images and preprocessed all data. We extensively researched and studied the newest types of models which could be possibly utilized for the image synthesis. Afterwards, we chose and trained four different types of generative models including Deep Convolutional GAN, Progressive Growing GAN, StyleGAN and Generative Adversarial Transformer. We performed the evaluation of the models both quantitatively and qualitatively for the comparison of their performance. Overall, models trained on the cropped images exhibited significantly superior performance compared to models learned from the dataset of padded images. The batch containing several fake images was generated for further qualitative assessment by dental experts. Feedback from professionals provided us with valuable information about the performance

of trained models. StyleGAN model has shown the most impressive results out of all mentioned models since dental professionals were not able to identify the generated images from the real ones. The StyleGAN model achieved also the best values in quantitative evaluation having the lowest FID scores out of all architectures (see Table 6.1).

Further work on the topic might involve proceeding with training the networks to generate images of higher resolutions as well as hyperparameter tuning. Moreover, further evaluation would be required by dental experts before using the synthetic images in practice.

Bibliography

[1] Charu C Aggarwal et al. Neural networks and deep learning. *Springer*, 10(978):3, 2018.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[3] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

[4] Jason Brownlee. *Generative adversarial networks with python: deep learning generative models for image synthesis and image translation*. Machine Learning Mastery, 2019.

[5] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038. IEEE, 1999.

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

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Gen-

erative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[8] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017.

[9] Paul S Heckbert. Survey of texture mapping. *IEEE computer graphics and applications*, 6(11):56–67, 1986.

[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018.

[11] Drew A. Hudson and C. Lawrence Zitnick. Generative adversarial transformers. *CoRR*, abs/2103.01209, 2021.

[12] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[13] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[14] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020.

[15] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Proc. NeurIPS*, 2021.

[16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[18] Kazuma Kokomoto, Rena Okawa, Kazuhiko Nakano, and Kazunori Nozaki. Intraoral image generation by progressive growing of generative adversarial network and evaluation of generated image quality by dentists. *Scientific reports*, 11(1):1–10, 2021.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[20] Walter YH Lam, Richard TC Hsung, Leo YY Cheng, and Edmond HN Pow. Mapping intraoral photographs on virtual teeth model. *Journal of dentistry*, 79:107–110, 2018.

[21] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[22] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[23] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[24] Andrew W Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander WR Nelson, Alex Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020.

[25] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[27] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.

Appendix

All scripts required for data acquisition, pre-processing and evaluation can be accessed at: https://github.com/DariaCarska/diplomova_praca/.