Artur Manuel Pascoal Ferreira

3D Lung Computed Tomography Synthesis using Generative Adversarial Networks

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
September 2021

Artur Manuel Pascoal Ferreira

3D Lung Computed Tomography Synthesis using Generative Adversarial Networks

Dissertation

Supervisor: Hélder Filipe Pinto de Oliveira
Co-supervisor: Tânia Pereira Lopes
Co-supervisor: Miguel Correira da Silva

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
September 2021

Resumo

Os modelos generativos têm vindo a aumentar em popularidade devido à sua habilidade de gerar novos dados, ao usar um *dataset* como referência. Tal é o caso com as *Generative Adversarial Networks* que são das arquiteturas mais usadas para sintetizar amostras de alta resolução. Exemplos do seu uso podem ser encontrados na área biomédica, para onde aplicações de *Deep Learning* têm vindo a ser rapidamente adaptadas. No entanto, estas requerem grandes e diversas quantidades de dados de pacientes para lidar com a inerente heterogeneidade de casos. Além disto, estudos de síntese frequentemente usam representações de dados em 2D, uma forma bastante limitada em informação quando se trata de tomografias computarizadas pulmonares, tendo em conta que patologias podem surgir em qualquer lugar no orgão. Deste modo, esta dissertação concentra-se na criação de um modelo capaz de gerar TACs volumétricos de pulmões com características realistas e espacialmente coerentes. Para tal, foi desenvolvida uma 3D PGGAN e treinada com o dataset LIDC, obtendo um resultado de 0.788 na métrica MS-SSIM e uma *accuracy* de 32% num dos testes visuais de Turing executado por profissionais de saúde.

Abstract

Generative models have been rising in popularity due to their ability to generate new data, by using the original dataset as a reference. Such is the case with Generative Adversarial Networks, which have been used to produce high-quality synthesized data. One of its use cases can be found in the biomedical domain, where Deep Learning applications are increasingly being adapted into. These, however, require diverse and large amounts of patient data to cope with the inherent data heterogeneity. Furthermore, synthesis studies often rely on a 2D representation of data, a very limited form of information when it comes to lung CT scans given that pathologies can manifest anywhere in the organ. As such, this thesis focuses on the development of a model capable of generating volumetric CT scans of lungs with realistic and spatially coherent features. A 3D-PGGAN was developed and trained on the LIDC dataset achieving an MS-SSIM score of 0.788 and an accuracy as low as 32% in a Visual Turing Test executed by medical professionals.

Keywords: Deep Learning, Generative Adversarial Networks, Synthesized 3D Images, Medical Imaging, Computed Tomography, Lung CT scans, Visual Turing Test

This work is financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project POCI-01-0145-FEDER-030263.

Contents

List of Tables

List of Figures

Acronyms

AC-GAN Auxiliary Classifier GAN

CAD Computer-Aided Diagnostic

cGAN Conditional GAN

CNN Convolutional Neural Network

COPD Chronic Obstructive Pulmonary Disease

CPM Competition Performance Metric

CT Computed Tomography

DCGAN Deep Convolutional GAN

DCNN Deep Convolutional Neural Network

ELBO Evidence Lower Bound

FOR False Omission Rate

FID Fréchet Inception Distance

fMRI Functional Magnetic Resonance Imaging

GAN Generative Adversarial Network

HP Health Professional

HU Hounsfield units

IS Inception Score

KL Kullback-Leibler

LIDC Lung Image Database Consortium

LSTM Long Short Term Memory

MLP Multi-Layer Perceptron

MRI Magnetic Resonance Imaging

MS Mode Score

MS-SSIM Multi-Scale Structural Similarity

NPV Negative Predictive Value

NSCLC Non-Small Cell Lung Cancer

PGGAN Progressive Growing GAN

RGB Red, Green and Blue

RNN Recurrent Neural Network

SCLC Small Cell Lung Cancer

SEER Surveillance, Epidemiology, and End Results

SSIM Structural Similarity

Std Standard Deviation

SVM Support Vector Machine

VAE Variational Autoencoder

VTT Visual Turing Test

Chapter 1

Introduction

1.1 Context

One of the leading causes of death in the world is cancer, causing 9.6 million deaths in 2018 alone. In the same year, lung cancer was the most common type, both in terms of incidence and mortality, having 2.1 million new cases and 1.8 million deaths worldwide [1]. As 5-year survival rate is low [2], early diagnosis and screenings are critical to improve patient outcomes, as radiologists make use of Computed Tomography (CT) scans in order to detect and evaluate lung pathologies indicative of lung cancer. However, this method is prone to human error given that, for example, lung nodules can have varying shapes, sizes and textures, despite being benign or malignant masses. Moreover, CT slice thickness has also been shown to affect accuracy in detecting lung nodules [3]. Given this complexity, human evaluation can often lead to misdiagnosis.

Computer-Aided Diagnostic (CAD) systems are a tool that have been developed to assist medical professionals in the improvement of the diagnosis. As such, there has been much effort in developing these systems as a way to reliably identify and characterize lung conditions in CT scans. Hussein, et al. [4] demonstrated promising results by using a 3D-Convolutional Neural Network (CNN) to classify the properties of lung nodules: malignancy, calcification, lobulation, sphericity, speculation, margins and texture. Although a 91.2% accuracy was achieved, the dataset used consisted of only 1018 CT scans, which can be considered a small data size especially given that deep learning models need large amounts of data. However, medical data is very hard to come by, with it frequently having fees and the need of approval by an ethical committee to gain access. This is emphasized in the case of lung imaging data where in order to identify and classify pathologies, annotated data by experts is needed, but that can be very expensive and time consuming.

To overcome the problem of scarce data, generative models have been used as a data augmentation method because of their ability to synthesize new data [5]. Recently, three types of generative models have been used for image synthesis: Autoregressive models [6], Variational Autoencoders (VAEs) [7] and Generative Adversarial Networks (GANs) [8]. Autoregressive models train a

function that learns the probability distribution of each pixel given the pixels that came before. VAEs are models that learn to maximize a lower bound on the log-likelihood of data. And GANs consist of a two player minimax game (a generator and a discriminator), where the goal is to have the generator capture the distribution of real data and create synthetic samples that completely fool the discriminator. All of these come along with their pros and cons:

	Pro	Con
Autoregressive	High Quality Outputs	Inefficient to train
VAE	Easy to Train	Blurry outputs
GAN	High Quality Outputs	Hard to train

Table 1.1: Pros and cons of generative models.

In medical imaging generation, most studies consist of implementing a model capable of synthesizing scan slices or other 2D types of data [9]. However, healthcare professionals often make use of the 3D aspect of data, as is the case with lung CT scans where analyzing lung pathologies with a single slice can be difficult. Furthermore, creating a 3D CT scan by using many slices generated by a 2D model may result in spatial inconsistencies, making the final output seem unrealistic. In 3D medical imaging synthesis, most studies implement image-to-image translation for cross-modality generation [10–13] and volume in-painting [14, 15]. In [10], a 3D Cycle GAN [16] was used to synthesize Functional Magnetic Resonance Imaging (fMRI) volumes given T1-weighted volumes and vice-versa. This versatility of the Cycle GAN opens a path for interesting applications since it can produce two types of scans with only one model, however, this comes at the price of computational cost as the model requires high-end equipment to train it. Additionally, the evaluation of the outputs is only made perceptually by the reader of the study, having no objective way to measure the overall quality of the model. In [14], a 3D Conditional GAN (cGAN) was used for in-painting lung nodules given a volume of interest with the nodules erased. It's shown that such an approach can improve lung nodule segmentation, however, the study restricts itself to a view of the lung surrounding lung nodules, disregarding any other lung pathology. Other studies use a noise vector as input instead of a volume, like in [17], where a 3D Progressive Growing GAN (PGGAN) was used to generate new T1-weighted brain volumes that don't represent any specific person. Although the outputs of the model look very appealing, again, the only evaluation performed is made by the reader. As such, is it reasonable to assume that the field of volumetric synthesis in medical imaging still has much room to grow.

1.2 Motivation

Currently, most studies in CT lung scan synthesis rely on a 2D representation of data [9], which is very limited in information. Although there are studies that explore the generation of 3D CT lung scans, most are focused solely on the synthesis of specific characteristics like lung nodules [15], and/or lack any form of evaluation. In fact, pulmonary nodules can be useful in detecting lung cancer [18] however, this condition can be a consequence of many other pathologies that

can be found anywhere in the lungs, like Chronic Obstructive Pulmonary Disease (COPD) [19]. Additionally, as it's common that medical imaging data isn't widely and freely available, being able to generate this type of data without restrictions could attract and help develop the overall research around CAD systems.

1.3 Goals and Contributions

The goal of this work is to research and develop a generative model, as well as evaluate it through quantitative metrics and clinical evaluation. This model should be capable of synthesizing 3D CT scans of lung images with high resolution and with realistic, spatially coherent characteristics. Furthermore, synthesized samples from the model should be capable of fooling a medical professional to think that it is, in fact, a real scan through a Visual Turing Test (VTT).

Additionally, the contributions of this work are the following:

- Test the current limitations of 3D medical data synthesis;
- Development of an efficient method for the synthesis of 3D CT lung scans;
- Exploration of metrics for the evaluation of 3D generation models and data.

1.4 Structure

The dissertation consists of 6 chapters, including this one. To start, chapter 2 is a study of the lungs including its anatomy, imaging techniques, main pathologies and its respective cancer. Chapter 3 contains an overview of generative models, their uses in 3D medical imaging and commonly used evaluation metrics. Chapter 4 describes the datasets used, their respective pre-processing pipeline, the model's architecture and its training details. Chapter 5 presents the obtained results and their evaluation. Finally, chapter 6 concludes the document by commenting on achievements and future work.

Chapter 2

Medical Background

In order to develop a model capable of synthesizing lung Computed Tomography (CT) scans, there has to be a good understanding of the lungs and their characteristics. As such, in this chapter first, the lungs anatomy will be analyzed, followed by its respective cancer and some associated pathologies. Finally, it is reviewed how CT scans are produced and some advantages of their use.

2.1 Lung Anatomy

The lungs are a paired cone-shaped organs lying in the thoracic cavity separated from each other by the heart and other structures in the mediastinum, with each one having a base resting on the diaphragm[1]. Each lung is invested in a serous pleural sac consisting of two membranes: the pulmonary pleura that invest the lungs; the parietal pleura that line the pulmonary cavities and adhere to the thoracic wall, mediastinum and diaphragm. Since the heart tilts to the left, the left lung is smaller and lighter than the right and has an indentation, called the cardiac impression, to accommodate the heart.

Each lung is composed of a set of lobes, divided by fissures, with the right lung containing three: upper, middle and lower lobe. On the other side, the left lung contains only two: the upper and lower lobe. Additionally, each lobe is divided into a series of bronchopulmonary segments. The right lung contains ten: apical, posterior and anterior in the superior lobe; lateral and medial in the middle lobe; superior, anterior, medial, lateral and posterior in the inferior lobe. The left lung contains eight: apicoposterior, anterior, superior and inferior in the superior lobe; superior, anteromedial, lateral and posterior in the inferior lobe [20].

When breathing, air goes through the trachea and gets partitioned into channels called bronchi, which themselves get divided into smaller branches, the bronchioles[2]. At the end these structures can be found the alveoli, the sacs that absorb the oxygen from the inhaled air are remove carbon

[1] Phisiopedia, Lung Anatomy, 09-09-2021

[2] American Cancer Society, What is lung cancer?, 09-09-2021

dioxide from the blood (Figure 2.1).

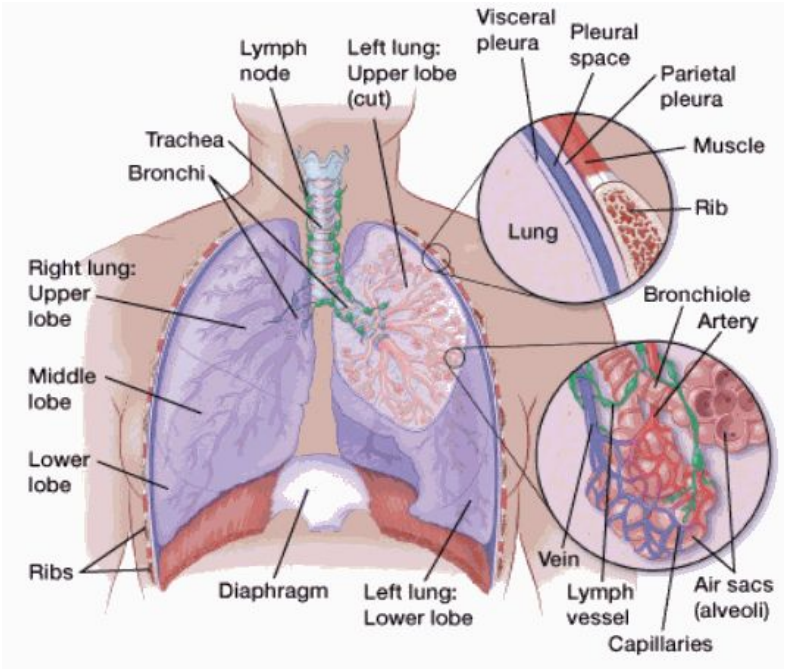


Figure 2.1: Lung Anatomy[3].

2.2 Lung Imaging

A CT scan is a non-invasive medical imaging procedure that produces cross-sectional images of the body[4] and is based on the principle that different tissues (e.g. muscle, bone) reflect and absorb X-rays differently. To capture the images, a motorized table moves the patient through a circular opening in the system, while an X-ray source and a detector rotate around the patient, taking snapshots (Figure 2.2).

In each rotation, these snapshots are sent to a computer to reconstruct a 2D view of the scanned area, named slice (Figure 2.3), which can be stacked along with other images to make a 3D CT volume. Occasionally, the procedure involves the intravenous administration of a contrast agent to highlight a specific area inside the body.

CT scans can be used for many purposes like guidance for tests or treatments (e.g. surgeries) or the detection and monitoring of certain conditions (e.g. emphysema). This method has the advantage of being painless and low-risk since the amount of radiation produced by the X-rays doesn't carry significant risk[6]. Additionally, CT scans are cheaper and faster than other 3D scans, like Magnetic Resonance Imaging (MRI) scans, making them the most common in lung

[3]American Cancer Society, What is lung cancer?, 09-09-2021
[4]Physiopedia, CT Scans, 09-09-2021
[5]Thoratic Key, Basic Principles in Computed Tomography (CT), 10-09-2021
[6]HealthyWA, Radiation risks of X-rays and scans, 15-09-2021

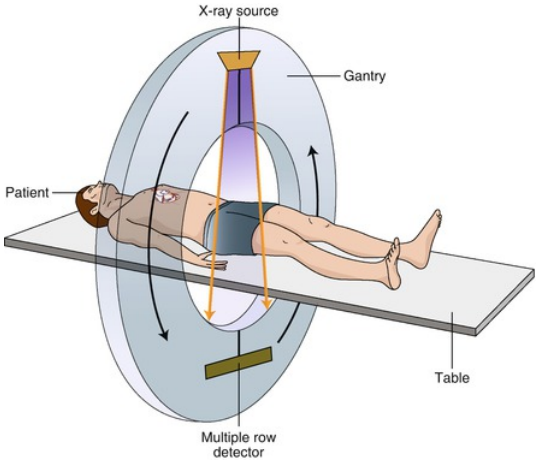


Figure 2.2: Diagram of a Computed Tomography scanner[5].

Figure 2.3: Example of a thoracic CT slice [21].

cancer screenings[7].

2.2.1 Chronic Obstructive Pulmonary Disease

Chronic Obstructive Pulmonary Disease (COPD) is the name given to the group of lung conditions that cause breathing difficulties[8]. Although it can be found in people who never smoked, the likelihood of developing the disease is smoking or exposure to harmful fumes. Breathing problems tend to worsen over time and despite existing treatment, damages caused to the lungs by this condition are permanent, making it extremely difficult to deal with. Moreover, its most common subtypes are chronic bronchitis and emphysema, which usually occur together.

[7]MSD Manual, Chest Imaging, 15-09-2021
[8]National Health Service, Chronic obstructive pulmonary disease (COPD), 09-09-2021

Chronic bronchitis occurs when the lungs' airways, the bronchi, become inflamed and irritated[9]. This occurrence causes mucus to build up and makes it difficult for the lungs to move air through its airways. Furthermore, it has been shown that the presence of chronic bronchitis is associated with 1.6 to 2.5 times of increased risk of lung cancer [22].

Emphysema is mainly caused by smoking or exposure to airborne toxins[10]. Initially, it may not show any symptoms, being only visible through CT scans (Figure 2.4). It is a progressive condition, that causes the air sacs in the lungs to become damaged or even destroyed. As the disease progresses, the lungs become unable to absorb oxygen and release carbon dioxide, causing shortness of breath. Additionally, studies have found that emphysema possibly increases the likelihood of lung cancer by 3.8 times [23], making the early diagnosis of this condition a possible way to detect lung cancer.

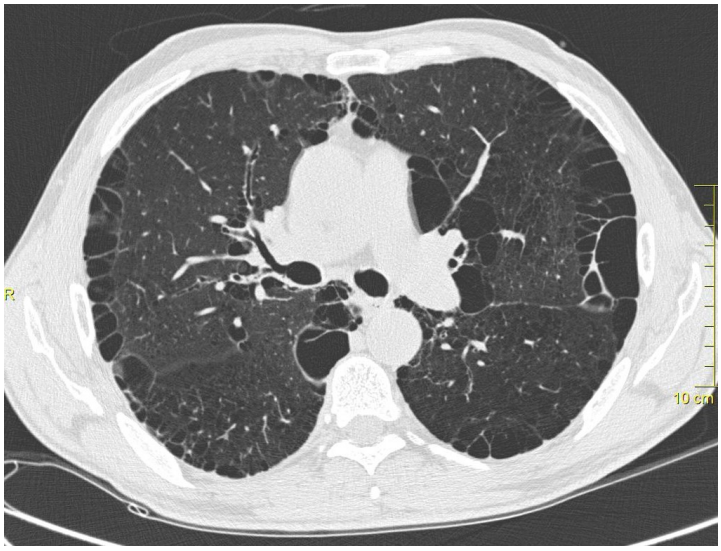


Figure 2.4: CT scan of a lung with emphysema[11].

2.3 Lung Cancer

According to the National Cancer Institute[12], cancer is a disease that is caused by changes to the genes that control the behavior of the cells, in particular to how cells should grow and divide. This results in some body cells dividing without stopping, creating masses of tissue called tumors. Unlike benign tumors, malignant tumors can spread or invade other regions of the body through the blood or the lymph system, forming new tumors far away from the original location.

There are two main types of lung cancer: Non-Small Cell Lung Cancer (NSCLC) and Small Cell Lung Cancer (SCLC). According to the American Cancer Society[13], NSCLC is the most

[9] MedlinePlus, Chronic Bronchitis, 09-09-2021
[10] Harvard Health, Emphysema, 09-09-2021
[11] Radiopedia, Paraseptal emphysema and subpleural bullae, 09-09-2021
[12] National Cancer Institute, What Is Cancer?, 09-09-2021
[13] American Cancer Society, What Is Lung Cancer? , 09-09-2021

common type with about 80-85% of cases. Its subtypes are:

- **Adenocarcinoma**: starts in cells that secrete mucus, is usually found in the outer regions of the lung and is the most common type of lung cancer;
- **Squamous cell carcinoma**: starts in cells that line the airways of the lungs, is often found in the central part of the lungs and is often related to a smoking history;
- **Large cell (undifferentiated) carcinoma**: appears in any part of the lung and tends to grow quickly, which can make treatment harder to succeed.

As for the SCLC type, this category represents 10-15% of all lung cancer cases. It usually begins in the bronchi, the airways leading to the lungs, and then quickly grows and spreads through the organ. This cancer tends to develop faster than the NSCLC type and often recurs in people that develop it[14].

The Surveillance, Epidemiology, and End Results (SEER)[15] program, tracks 5-year survival rates for both NSCLC and SCLC in the United States based on how far the cancer has spread, where cases are grouped into three categories:

- **localized**: there is no sign of cancer outside of the lung (17% of cases);
- **regional**: cancer has spread into nearby structures (22% of cases);
- **distant**: when cancer has spread to other parts of the body (57% of cases).
- **unknown**: when the cancer stage is not known (4% of cases).

Between 2010 and 2016, lung cancer's 5-year survival rate was 59.0%, 31.7% and 5.8% and 8.3% for localized, regional, distant and unknown stages respectively. This indicates that detecting lung cancer at an early stage has a significant impact to the chance of surviving.

2.3.1 Pulmonary Nodules

A lung nodule is a rounded, well or poorly defined, growth that is up to 3 cm in diameter [24] (Figure 2.5). Structures with a larger size are classified as masses, due to the likelihood of being malignant.

A lung nodule can originate from infections, scar tissue or tumors and is not necessarily malignant. Even so, identifying the malignancy of a nodule as fast a possible is crucial. If detected at an earlier stage, the survival rate is higher, having a 60-80% 5-year survival rate in stage I NSCLC, and more options for treatment are available [25].

[14]Cancer Treatment Centers of America, Lung Cancer Types, 09-09-2021
[15]Surveillance, Epidemiology, and End Results, Cancer Stat Facts: Lung and Bronchus Cancer, 09-09-2021

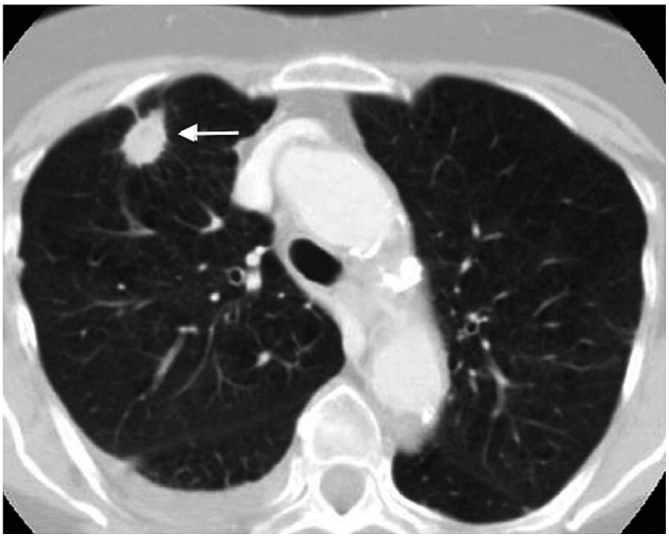


Figure 2.5: A thoracic CT slice with a lung nodule (arrow) [21].

According to the results of the National Lung Screening Trial, a United States based lung cancer screening trial [26], the diameter of a pulmonary nodule or mass was directly related to its malignancy (Table 2.1).

Diameter	PPV
4-6mm	0.5
7-10mm	1.7
11-20mm	11.9
21-30mm	29.7
>30mm	41.3

Table 2.1: The Positive Predicted Value (PPV) is defined as the proportion of patients with confirmed lung cancer among those with a positive result on screening whose lung-cancer status was known [26].

Furthermore, when examining a CT scan, to evaluate the likelihood of the malignancy of a nodule, other features like shape, calcification, intranodular fat, ragged margins, and others are taken into consideration [25].

2.4 Summary

Lung cancer is a disease that develops and spreads to other regions of the body quickly since most cases are discovered at a later stage. Moreover, the 5-year survival rate drops significantly with the expansion of cancer, making the early diagnosis a critical aspect of the outcome of the patient. As for indicators of the decease, pulmonary nodules are a good reference, being that their size is related to the likelihood of malignancy and the presence of COPD has also been shown to increase the probability of cancer. The detection and analysis of lung cancer are usually

made through CT scans as it is a low-cost procedure, has faster scan times than the alternatives and provides a 3D view of the region.

Chapter 3

Literature Review

Generative models are a subset of unsupervised learning frameworks that, when trained with samples from a certain distribution, can generate new samples from that same distribution [27]. The main three types of generative models are presented in this chapter: Autoregressive models, Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs).

Although this work focuses on the synthesis of 3D data, most of the models shown in this chapter are trained with and generate 2D data. However, the methods themselves are analyzed, and in most cases, these can be adapted to a 3D version.

3.1 Autoregressive models

Autoregressive models implicitly learn the distribution of each sequence given the ones that came before, using the Chain Rule for Conditional Probability. The PixelRNN [6] model, implements a Recurrent Neural Network (RNN) that generates images pixel by pixel, instead of the whole image at once. Formally, the goal is to assign a probability p(x) to each image $\mathbf{x}$ formed of $n \times n$ pixels and is defined as:

$$p(x) = \prod_{i=1}^{n^2} p(x|x_1, ..., x_{i-1}) \tag{3.1}$$

where the probability for the i-th pixel is calculated given all previous pixels $x_1, ..., x_{i-1}$ and each color of the color channels, Red, Green and Blue (RGB).

For the study [6], two types of Long Short Term Memory (LSTM) layers were used: Row LSTM and Diagonal BiLSTM (Figure 3.1). The Row LSTM is a unidirectional layer that processes an image row by row and top to bottom, with each row being computed at once and each pixel having as context the triangular area above it. On the other hand, the Diagonal BiLSTM layer uses the entire context that came before by scanning the image in a diagonal fashion. The results showed that the Diagonal BiLSTM achieved the best test set performance between the two.

Another autoregressive variant, the PixelSNAIL [28], makes use of both convolution and self-

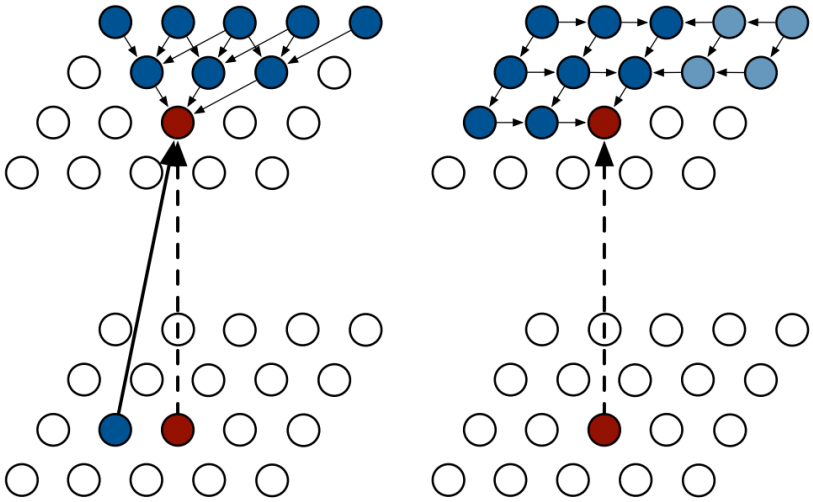


Figure 3.1: Row LSTM (left) and Diagonal BiLSTM (right) layers [6]. The Row LSTM uses the triangular area above it as context to generate new pixels, while BiLSTM uses all the context of the area above that comes before the pixel itself.

attention techniques. Causal convolutions models apply convolutions over a sequence, offering high bandwidth access to earlier parts of it, but have a limited context size with far away elements having little effect on the result. By contrast, self-attention models allow full access to the context, but only offer pinpoint access to small amounts of information. By combining these two, a model can have high bandwidth access without constraints on the amount of information it can use. The results obtained showed that the PixelSNAIL model outperformed all other autoregressive models, including the PixelRNN.

3.2 Variational Autoencoders

Autoencoders [29] consist in combining two neural networks, an encoder and decoder, trained in an unsupervised setting. In the middle of the networks exists the bottleneck region, where the network encodes to and decodes from. It is possible to use an autoencoder for data generation, however, because it is only trained to have as little loss possible, the latent space is not regularized and non-continuous. VAE [7] (Figure 3.2) aim to overcome this by adding a Bayesian component that learns the parameters representing the probability distribution of the data, which is achieved by imposing a prior on the probability of the input [30]. The model is then trained as follows:

- Encode the input as a distribution over the latent space;
- Sample a point from the latent space according to that distribution;
- Decode the sampled point and calculate the reconstruction error;
- Backpropagate the reconstruction error.

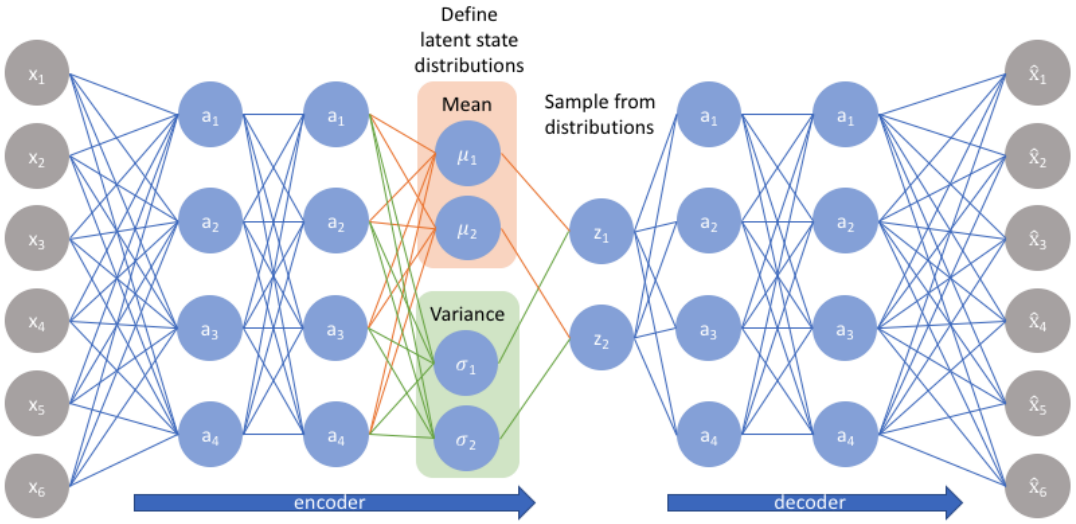


Figure 3.2: Architecture of Variational Autoencoder[1]. The input is encoded as distribution (Encoder), the Latent Space samples from that distribution and the latent vector is then decoded (Decoder).

The encoder can then be defined by $p(z|x)$, where z is the encoded variable, x the decoded one and can be calculated using the Bayes rule:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz} \tag{3.2}$$

As the integral in the denominator is an intractable distribution, this value is estimated through variational inference. The value of $p(z|x)$ is instead an approximate value using another distribution $q(z|x)$ that is tractable, such as the Gaussian distribution. Subsequently, the Kullback-Leibler (KL) divergence is used to evaluate the difference between both distributions, being that the goal is to minimize it:

$$minKL(q(z|x)||p(z|x)) = \mathbb{E}_q[\log q(z|x)] - \mathbb{E}_q[\log p(x,z)] + \log p(x) \tag{3.3}$$

Since there still exists an intractable term, $\log p(x)$, the Evidence Lower Bound (ELBO) function can be used instead:

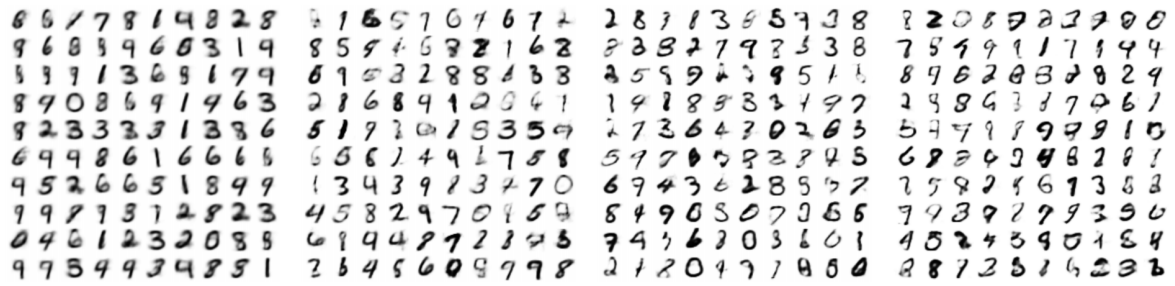
$$ELBO = \mathbb{E}_q[\log q(z,x)] - \mathbb{E}[\log q(x|z)] \tag{3.4}$$

With this, the loss function can then be defined as the reconstruction loss combined with the ELBO:

$$min\mathcal{L}(x,x') + ELBO \tag{3.5}$$

Although samples obtained with VAE are realistic and understandable, still, one of its most significant drawbacks are blurry outputs, as show in Figure 3.3.

[1]Jeremy Jordan, Variational Autoencoders, 2018



(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

Figure 3.3: Random samples generated by a VAE trained with the MNIST dataset at different dimensionalities of latent space [7].

3.3 Generative Adversarial Networks

First proposed by Goodfellow, et al. [8], GANs generally consist of two models, a generator G and a discriminator D, in an adversarial environment (Figure 3.4).

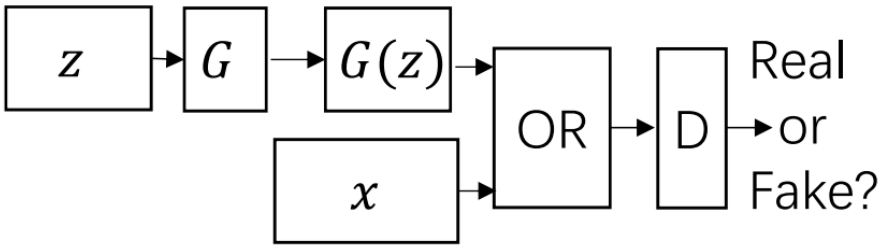


Figure 3.4: The GAN structure consists of two models, G and D, where the generator tries to create data that the discriminator is unable to distinguish from real samples [31].

While G tries to capture the data distribution, D tries to discriminate samples generated by G from real ones. During training, both models improve until G is able to generate samples that are indistinguishable from the real data. Formally, the objective of G is to learn the mapping of $p_z(z)$ to $p_{data}(x)$, where z is a sample from a latent space and x is a real sample. The goal of D is to learn the mapping of data to the probability of it being real (closer to p_{data}, which means higher probability) or generated (closer to p_z, which means lower probability).

The process of training the models can be described as a two-player minimax game, with the value function $V(G, D)$ being:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log 1 - D(G(z))] \tag{3.6}$$

In practice, Equation 3.6 does not work well early in training. This is due to D being able to distinguish with high confidence between real and generated data, while G is poor. To solve this, rather than training G to minimize $\log(1 - D(G(z)))$, G is trained to maximize $\log D(G(z))$. In other words, G maximizes the probability of D being mistaken instead of minimizing the probability of D being correct.

3.3.1 Deep Convolutional GAN

In the original GAN [8], both the generator and the discriminator are Multi-Layer Perceptron (MLP) networks. To improve the performance of GANs in images, the Deep Convolutional GAN (DCGAN) [32] uses Deep Convolutional Neural Networks (DCNNs) and is now the basis for many GAN architectures [31]. This approach was based in adopting and modifying three changes to Convolutional Neural Network (CNN) architectures:

- Using an all convolutional network, replacing pooling layers with strided convolutions;
- Remove fully connected layers;
- Use Batch Normalization to stabilize learning.

To validate the model, the authors used the DCGAN as a feature extractor for a supervised dataset and evaluated the performance of a Support Vector Machine (SVM) fitted on top of those features. The discriminator's convolutional features are used by max pooling, flattening and concatenating each layer forming a vector which is then used to train the SVM. Through this method was obtained an accuracy of 82.8%, outperforming K-means based techniques.

Additionally, it is shown that the generator of the DCGAN contains vector arithmetic properties meaning that, combining two or more latent space codes, allows for the manipulation of semantic qualities of the generated samples, as shown in Figure 3.5.

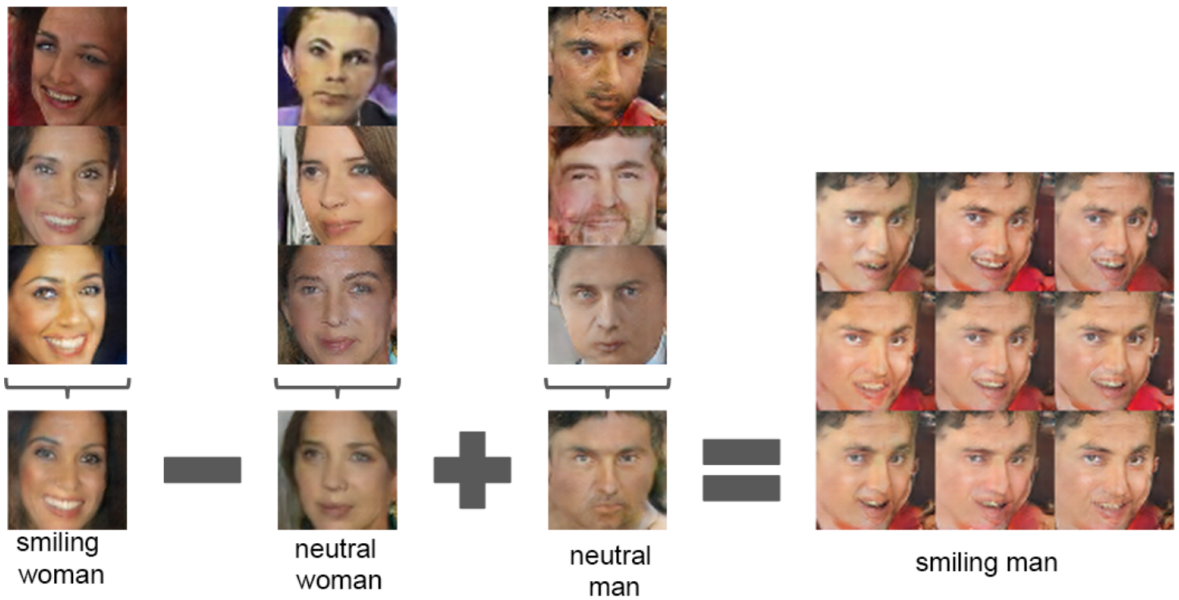


Figure 3.5: Latent Vector arithmetic's using DCGAN [32]. By combining different latent codes and performing operations, it is possible to manipulate the generator to obtain certain results.

3.3.2 Conditional GAN

Introduced by Mirza, et al. [33], the Conditional GAN (cGAN) extends the original GAN architecture to a conditional model by adding an input layer to both the discriminator and the generator (Figure 3.6).

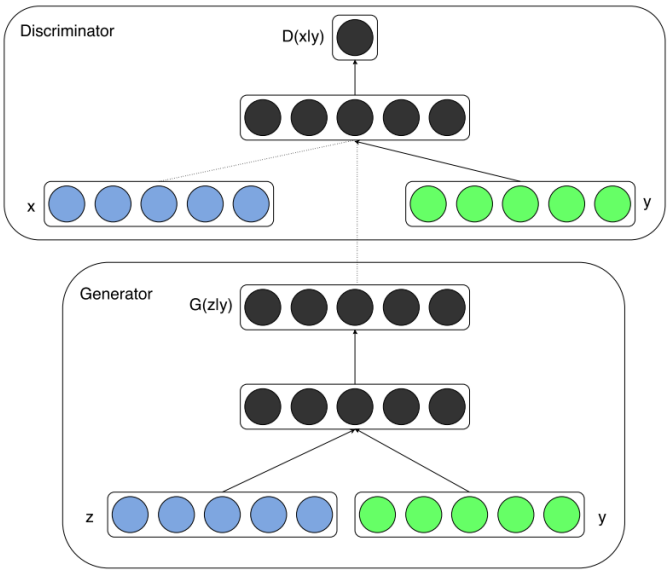


Figure 3.6: cGAN structure [33]. Both the generator and discriminator use x or z together with a label y.

This architecture has advantages like having higher quality generated samples for labeled data and being able to control the output of the generator by conditioning it (Figure 3.7). The value function is changed with it now being:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)] + \mathbb{E}_{z\sim p_z(z)}[\log 1 - D(G(z|y))] \tag{3.7}$$

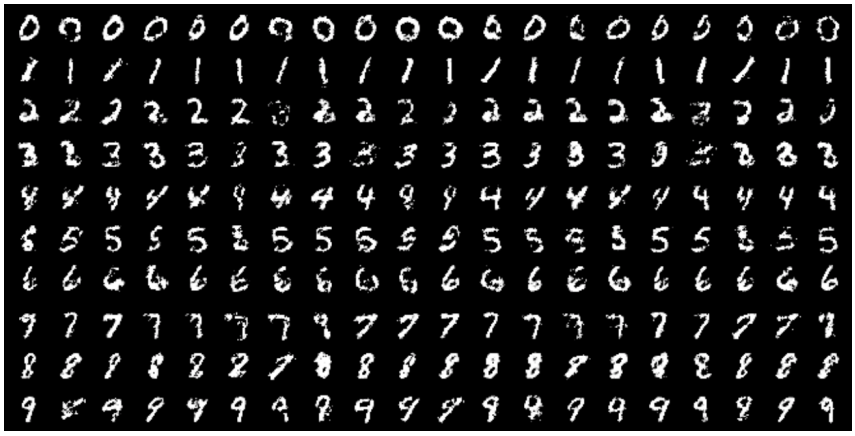


Figure 3.7: Generated MNIST digits using cGAN with each row conditioned on one label [33].

3.3.2.1 Auxiliary Classifier GAN

Proposed by Odena, et al. [34], the Auxiliary Classifier GAN (AC-GAN) also improves on the original GAN architecture by conditioning it. Like the cGAN (Section 3.3.2) the generator is given a latent code as well as a class label as input. On the contrary, the discriminator is tasked to classify the label of the data, instead of using it as input. The loss function is changed, with it being the traditional loss (Equation 3.6) with an added term L_C, the log-likelihood of the correct class label:

$$L_C = \mathbb{E}[\log P(C = c|X_{real})] + \mathbb{E}[\log P(C = c|X_{fake})] \tag{3.8}$$

As for the evaluation of the model, the Multi-Scale Structural Similarity (MS-SSIM) (Section 3.6.5) metric was used by measuring scores between 100 randomly chosen pairs of images for a given class of the ImageNet dataset [35]. The mean score for generated samples 0.18 with a standard deviation of 0.08, indicating high diversity in outputs.

3.3.2.2 Pix2pix

Proposed by Isola, et al. [36], pix2pix is an image-to-image translation framework using cGANs. Here, the goal G is to learn the mapping of an input image x to an output image G(x) of a different domain, therefore, training requires pairs of data. As for D, it uses both x and G(x) or the real image y as its input to learn to distinguish real images from fake ones, as shown in Figure 3.8.

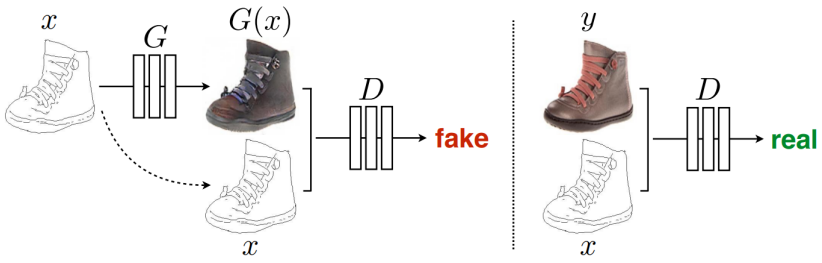


Figure 3.8: pix2pix structure [36]. Both the discriminator and generator take the map edges as input, with the discriminator taking pairs of images as input.

The value function of this model is similar to the cGan with the addition of the L_1 distance loss. The L_1 loss was used instead of $L2$ since the authors found that the latter produced blurrier results. The final function is:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log 1 - D(x, G(z, x))] + \lambda \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1] \tag{3.9}$$

where λ is a hyper-parameter controlling the L_1 distance term importance. In practice, z is not used as input of G, which leads to deterministic outputs. Instead, noise is provided to the model in form of dropout in both train and test times.

In the original paper [36], G followed a U-Net [37] structure in order to give the generator a way to circumvent the loss of low-level features that traditional encoder-decoder structures have, and image translation problems usually require. As for D, instead of using a standard convolution classifier, it was noted that classifying the whole image at once resulted in blurry samples, and so, developed a PatchGan which penalizes structure at the scale of patches. The PatchGan classifies if each of the $N \times N$ patches is real or fake, which results in notably less blur, as shown in Figure 3.9.

Figure 3.9: PatchGAN samples with varying patch sizes [36]. As patch size increases, the generator is forced to output more realistic results.

For evaluation, the authors used the FCN-8s [38] model architecture and trained it on the cityscapes dataset [39] for semantic segmentation, as well as the pix2pix model. The synthesized samples were then scored by the classification accuracy against the labels of the original photo. The best scores obtained consisted of 66% in per-pixel accuracy and 23% in per-class accuracy for the model conditioned by labels and with L1 loss, which also produced the higher quality results (Figure 3.10).

Figure 3.10: Pix2pix results with different losses [36]. As the L1 loss leads to blurry results and the cGAN to sharp results, by combining both, image quality is improved.

3.3.3 Cycle GAN

First proposed by Zhu, et al. [16], the Cycle GAN is an image-to-image translation framework where, unlike the pix2pix model, there is no need for paired data. In others words, while the goal of the model is to translate data from one modality to a different one, the training data from both modalities does not have to be related, which makes the creation of datasets much easier and the model versatility much greater.

The model contains two generators (Figure 3.11): G, which learns to map samples from domain X to domain Y and F, that translates data from domain Y to domain X. It also contains two adversarial discriminators: D_Y checks the probability of samples generated by G being real or fake and D_X does the same for samples generated by F.

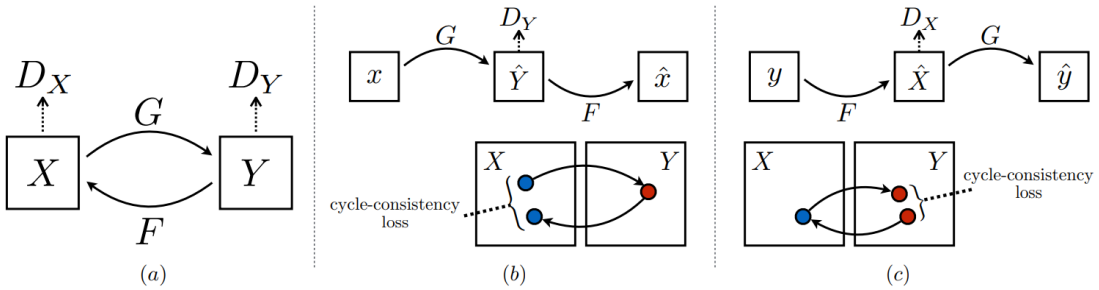


Figure 3.11: Cycle GAN structure [16]. The model contains two mapping functions: $G : X \rightarrow Y$ and $F : Y \rightarrow X$ (a). D_Y encourages G to translate X into domain Y, and vice versa for D_X and F (b). The mappings are regularized by checking the translation from one domain to another and back the sample should remain the same (c).

The loss function for the Cycle GAN is a mix of the traditional GAN loss (Equation 3.6), for both pairs of generators and discriminators, with a cycle consistence loss. For the generator G and its discriminator D_Y the loss is:

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}\big[\log D_Y(y)\big] + \mathbb{E}_{x \sim p_{data}(x)}\big[\log(1 - D_Y(G(x))\big] \tag{3.10}$$

The cycle consistence loss is the L_1 distance of an image and its generated sample for the other domain and is defined as:

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}\big[\|F(G(x)) - x\|_1\big] + \mathbb{E}_{y \sim p_{data}(y)}\big[\|G(F(y)) - y\|_1\big] \tag{3.11}$$

The loss function for the whole model is:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \mathcal{L}_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F) \tag{3.12}$$

where λ is a hyper-parameter controlling the L_1 distances term importance.

The evaluation method for this architecture is the same used in pix2pix (Section 3.3.2.2). The results obtained were similar, although worse, having 58% in per-pixel accuracy and 22% in per class accuracy.

3.3.4 Progressive Growing GAN

Based on Progressive Neural Networks [40], the Progressive Growing GAN (PGGAN) [41] proposes a new, but simple, training methodology for GANs. The process starts by training using low-resolution images, and progressively increasing the resolution by adding layers on the generator and discriminator simultaneously (Figure 3.12).

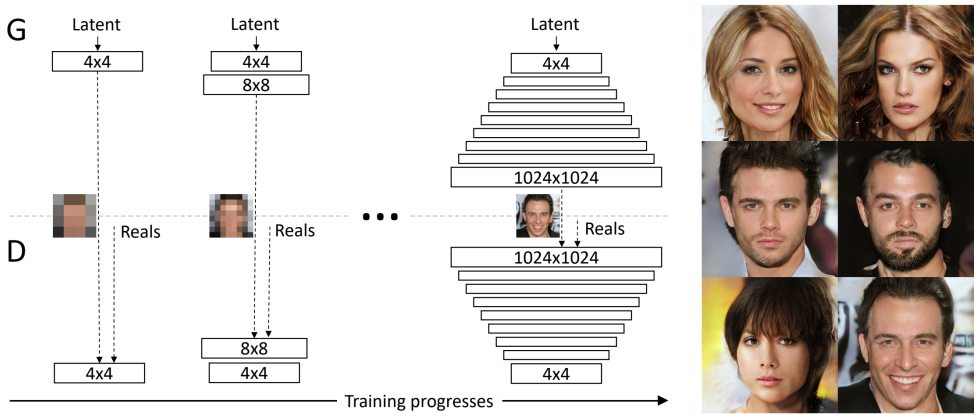


Figure 3.12: PGGAN training process [41]. The model starts by training with low resolutions images which get increased throughout time.

Initially, the model learns large-scale features of the image distribution and then incrementally shits its attention to finer details, instead of doing it all simultaneously. This results in a faster and more stable training process, a difficult achievement among GANs. When new layers are added they're faded in, meaning that initially, a new resolution layer is treated as a residual block, whose importance increases from 0 to 1 while training. In respect to the loss function, the WGAN-GP loss [42] was used and it is defined as:

$$\mathcal{L} = -\mathbb{E}_{x\sim p_{data}(x)}[D(x)] + E_{\tilde{x}\sim p_g}[D(\tilde{x})] + \lambda E_{\hat{x}\sim p_{\hat{x}}}[\left\|\nabla_{\hat{x}} D(\hat{x})\right\|_2 - 1)^2] \tag{3.13}$$

where $\hat{x}$ is sampled from both x and $\tilde{x}$:

$$\hat{x} = t\tilde{x} + (1-t)x \tag{3.14}$$

where t is uniformly sampled between 0 and 1.

For evaluation, the Inception Score (IS) (Section 3.6.1) was used with the CIFAR10 dataset, where the best score obtained was 8.80 with a standard deviation of 0.05, being at publication the best score among unsupervised methods.

3.3.4.1 Style GAN

Inspired by style transfer literature, the Style GAN [43] improves upon the PGGAN by modifying its architecture, with the most notable change being found in the generator. Unlike other models, the generator begins with a learned constant input, with the latent code being "injected" throughout each of its convolutional layers (Figure 3.13).

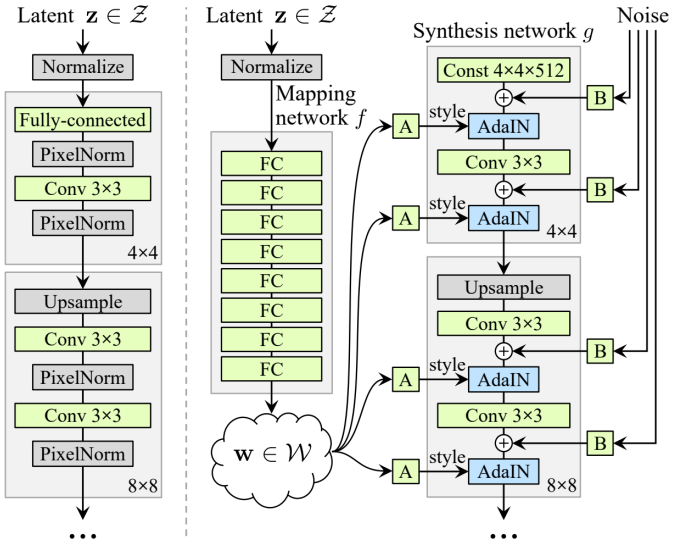


Figure 3.13: PGGAN generator (left) and Style GAN generator (right) [43]. The new design uses a non-linear mapping network from Z to W as a way to avoid an inconsistent latent space.

This novel idea allows for the control of the "style" at different scale levels of the output (since the architecture is still progressive) and even style-mixing. Having this ability, mixing styles is used as a regularization technique where a given percent of images are generated using two random latent codes, which prevents the network from assuming that adjacent styles are correlated. Additionally, this can be used to produce images from different styles while mixing them at various scales (Figure 3.14).

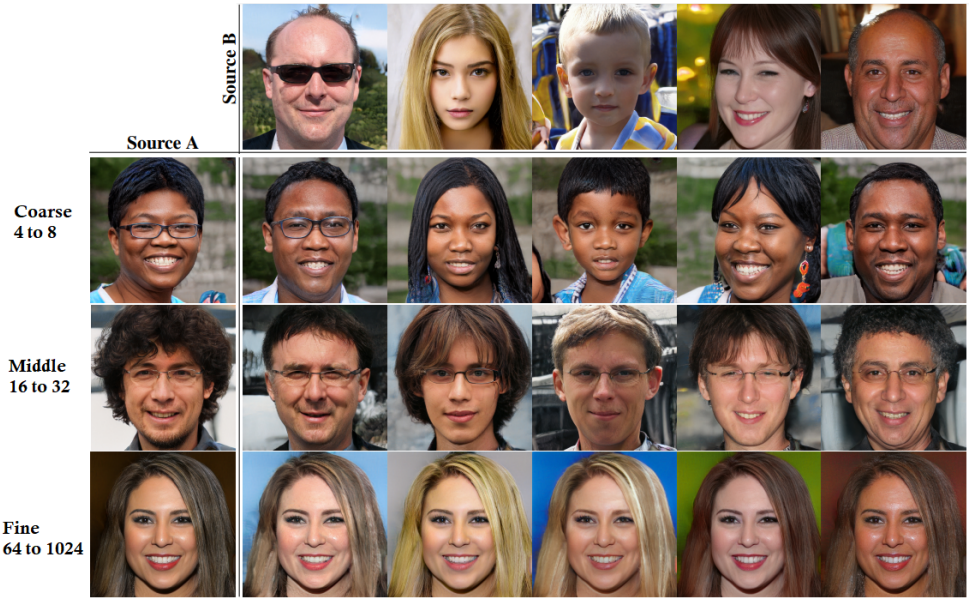


Figure 3.14: Mixing Styles at various scales (adapted from [43]). Defining the point at which the source B style is used can change the outputs, with being used in lower resolutions changing global features and in higher resolutions changing specific characteristics of the image.

As for evaluation, the authors made of use the Fréchet Inception Distance (FID) (Section 3.6.3)

metric using 50.000 images randomly sampled from the training set of both the CelebA-HQ [41] and the FFHQ [43] dataset, having scored 5.06 and 4.40, respectively.

3.3.4.2 Style GAN 2

Improving on the Style GAN [43], the Style GAN 2 [44] is a set of small changes to the architecture that improve results significantly. A notable difference came with weight demodulation (Figure 3.15), where the weights of a block are scaled to the incoming style (modulation) at the beginning of the block, and subsequently scaled by the L_2 norm (demodulation). This change is a more direct approach to normalization and results in fewer operations than in the original architecture where the weights were scaled and normalized to the style at the beginning of a block and normalized again at the end.

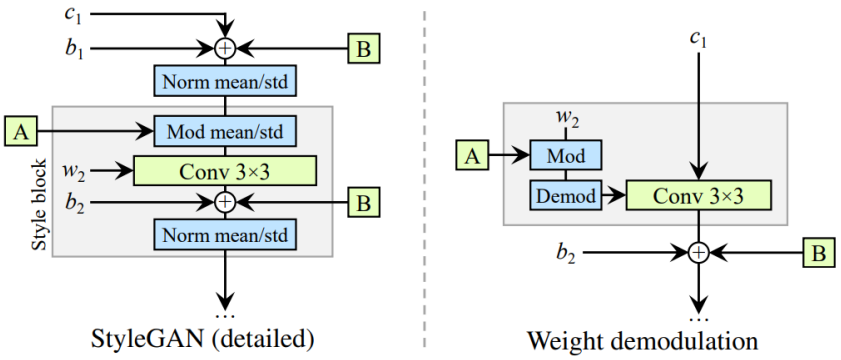


Figure 3.15: Generator blocks from StyleGAN (left) and StyleGAN 2 (right) [44]. The new block design uses fewer operations while having the same normalization effect on the convolutional filters.

Additionally, it also solves one of the visual problems of the outputs of the Style GAN, the water droplet artifact (Figure 3.16). This artifact is a systemic problem on Style GAN images caused by the generator sneaking signal strength through normalization when injecting the style into the weights.

Another change came with the decision of dropping the progressive growing nature introduced in the PGGAN [41] for a skip architecture. This is because while using a progressive growing model, as each resolution is momentarily the output resolution, the intermediate layers produce high-frequency outputs that can have too much importance in the final result. This behavior can impact the quality and natural look of the outputs, like producing "phase" artifacts (Figure 3.17).

Along with other changes, like increasing the number of feature maps in the later resolutions and regularizing the latent space using the Jacobian matrix between different latent codes, improve the FID score for the FFHQ dataset [43] to 2.84. These improvements not only make the Style GAN 2 outperform in metrics but perceptually as well when it comes to generating human faces, as shown in figure 3.18.

Figure 3.16: Images produced by the Style GAN contained water droplet artifacts [44].

Figure 3.17: Progressive growing leads to "phase" artifacts, like in this case where the teeth follow the camera and not the pose of the subject [44].

Figure 3.18: Hand-picked examples produced by Style GAN 2 [44].

3.4 3D GAN

Proposed by Wu, et al. [45], the 3D-GAN presented a novel way to generate 3D objects. The generator G has five volumetric fully convolutional layers and its goal is to learn a mapping from a 200-dimensional latent vector z to a 64^3 cube (Figure 3.19). The discriminator D however, classifies whether a cube was real or generated and its structure mostly mirrors the generator.

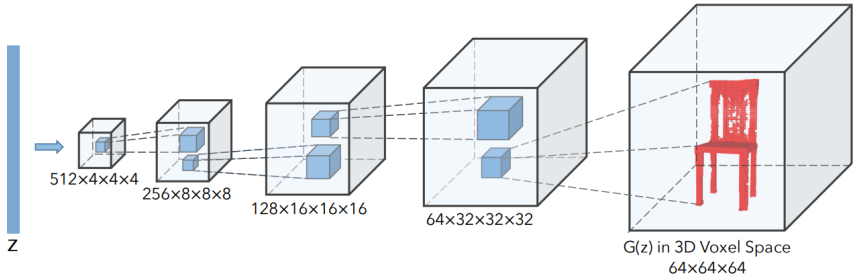


Figure 3.19: Generator of 3D-GAN [45]. The discriminator mostly mirrors the generator.

The loss function, follows the traditional adversarial loss (Equation 3.6), being:

$$L_{3DGAN} = \log D(x) + \log(1 - D(G(z))) \tag{3.15}$$

D tends to learn much faster than G since learning to generate objects in a 3D space is much harder than differentiating between real and synthetic ones, and so, an adaptive training strategy is implemented. The authors found that it helps to stabilize training if, for each batch, D only has its weights updated if the accuracy of the last batch is lower than 80%. Figure 3.20 shows some generated examples.

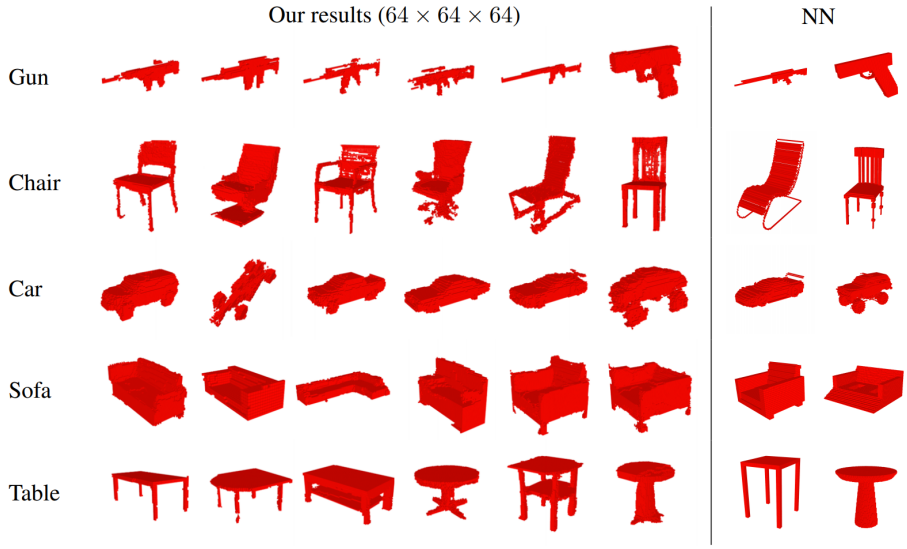


Figure 3.20: 3D-GAN generated examples. The last two objects in each row (NN) are the nearest neighbors from the training dataset [45].

Having trained the model conditionally, another way to explore the results is to show arithmetic

operations in the latent space. In this case, the authors used the functionality to show that new structures can be added to a specific type of furniture like for example, adding arms to a chair (Figure 3.21).

Figure 3.21: Arithmetic operations of the 3D-GAN. The left example shows how to add arms to a chair and the left example how to add layers to a table [45].

In the same paper, the authors also proposed the 3D-VAE-GAN, as an extension to the 3D-GAN. The idea is that if a 2D image has a mapping to the latent space, it is then possible to generate a 3D volume corresponding to it. This is made possible by using an additional encoder E, that learns the mapping of a 2D image to the latent representation z, which is then used as input on G. At this point, the model takes both volumes and images simultaneously, with the images being extracted from the 3D models in 72 different views (24 angles and 3 elevations). With this, the model was able to reconstruct volumes through images (Figure 3.22) of the IKEA dataset [46] with a mean voxel precision of 53.1%.

Figure 3.22: Reconstruction examples of the 3D-VAE-GAN [45].

3.5 3D GANs in Medical Imaging

As GANs are a framework capable of creating, translating and reconstructing new data, they have great potential in the context of medical imaging, being currently the most explored method in the space. In this section, state-of-the-art GANs used to synthesize volumetric data are analyzed, where first is presented noise-to-image approaches, followed by image-to-image translation models.

3.5.1 Noise-to-Image

In the field of neuro-imaging overall, few datasets exist and not with many samples. In an attempt to resolve this, in [47], a 3D Auto-Encoding Network was proposed in order to generate 3D brain MRI scans from a latent space. The model combines the advantages of being free of mode collapse from VAEs and having the reduced blur of GANs, by using both generative architectures with an additional code discriminator C (Figure 3.23).

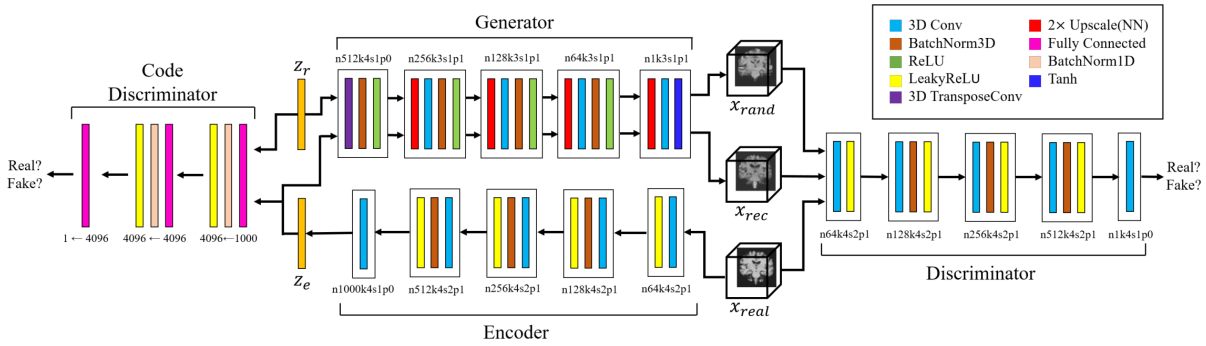


Figure 3.23: 3D Auto-Encoding network architecture [47]. The model uses a discriminator to distinguish codes being from the latent space or encoded, forcing real and fake distributions to sync.

C plays an adversarial game with the real sample encoder, and its goal is to distinguish between real encoded vectors and samples from the latent space. When it fails to do so, it means that the encoder is in sync with the latent space. The model produced realistic results (Figure 3.24) and in evaluation an MS-SSIM (Section 3.6.5) of 0.829 was obtained with the real samples having 0.846, indicating better diversity.

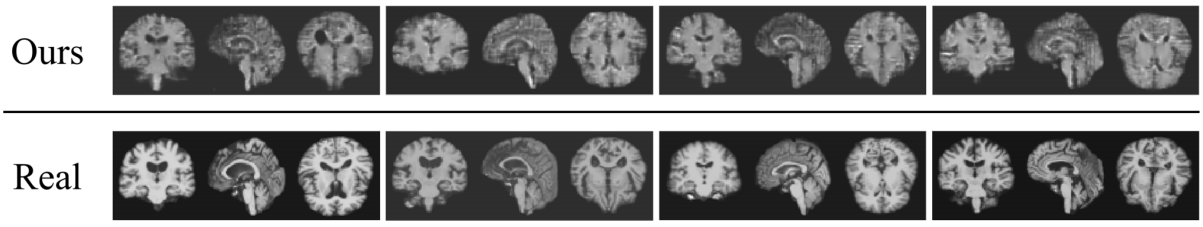


Figure 3.24: 3D Auto-Encoding GAN generated versus real samples [47].

Another method was proposed in [17], where a 3D-PGGAN was used to synthesize MR brain volumes from a latent space. The model takes the PGGAN architecture (Section 3.3.4) and adapts it to the 3D space by changing all 2D convolutions into 3D convolutions and reducing the number of convolutional filters for memory efficiency. The model starts by generating 4^3 volumes until training reaches the maximum resolution of 64^3, producing full 3D brain scans with perceptually good results (Figure 3.25), while unfortunately missing any kind of evaluation.

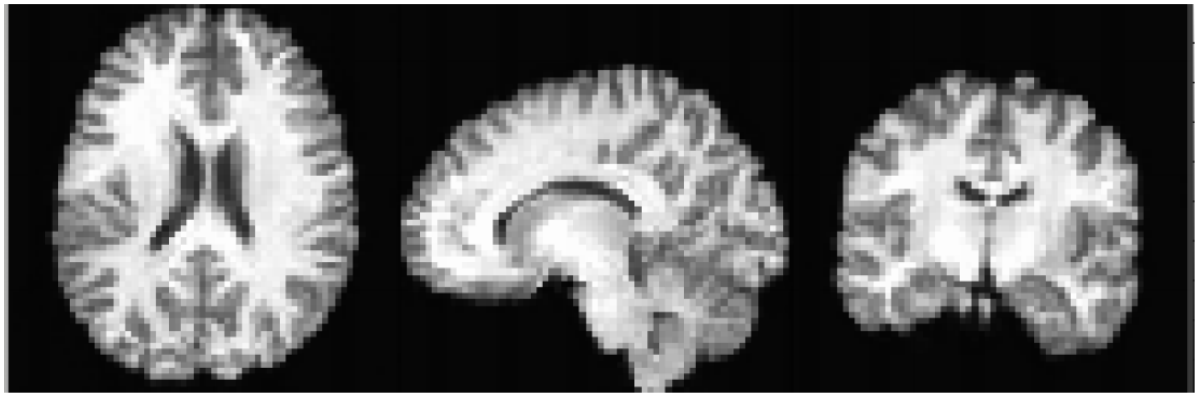


Figure 3.25: 3D-PGGAN generated example at a resolution of 64^3 upsampled to 128^3 [17].

3.5.2 Image-to-Image translation

Image-to-Image translation models in the medical imaging field are often related to cross-modality applications, however, some studies have used the approach for synthesis. Yang et. al. [48] proposed a model that could synthesize lung nodules in CT volumes, by using an in-painting approach. The architecture consists in using two generators instead of one (Figure 3.26), where the coarse generator receives the input volume of the nodule and its surroundings but with a mask applied to the nodule itself, accompanied by a class label to indicate if the nodule is malignant or benign. Its outputs are then fed to the refinement generator where, as the name suggests, the volume details are refined. Additionally, there are two discriminators, where one classifies the masked area exclusively (local discriminator) while the other classifies the whole volume (global discriminator).

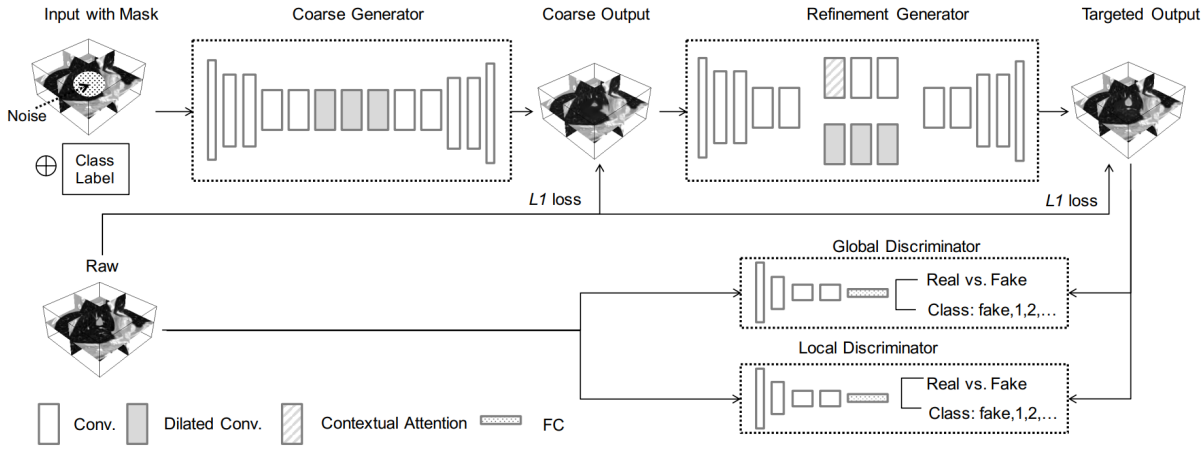


Figure 3.26: Lung nodule synthesizer architecture [48]. Unlike traditional GANs, this model includes two generators and discriminators. The extra generator is used to refine synthesized samples while the discriminators are used to evaluate features at different scales, locally and globally.

The model was evaluated using ResNets [49] trained on the LIDC dataset [21], with and without generated samples. The results show that accuracy, specificity and area under the curve were

higher while using the augmented dataset having, however, worst performance in sensitivity meaning that the model underachieved in classifying malignant nodules. The generated samples show that synthesized nodules tend to have smaller sizes than benign ones, as well as have better defined boundaries (Figure 3.27).

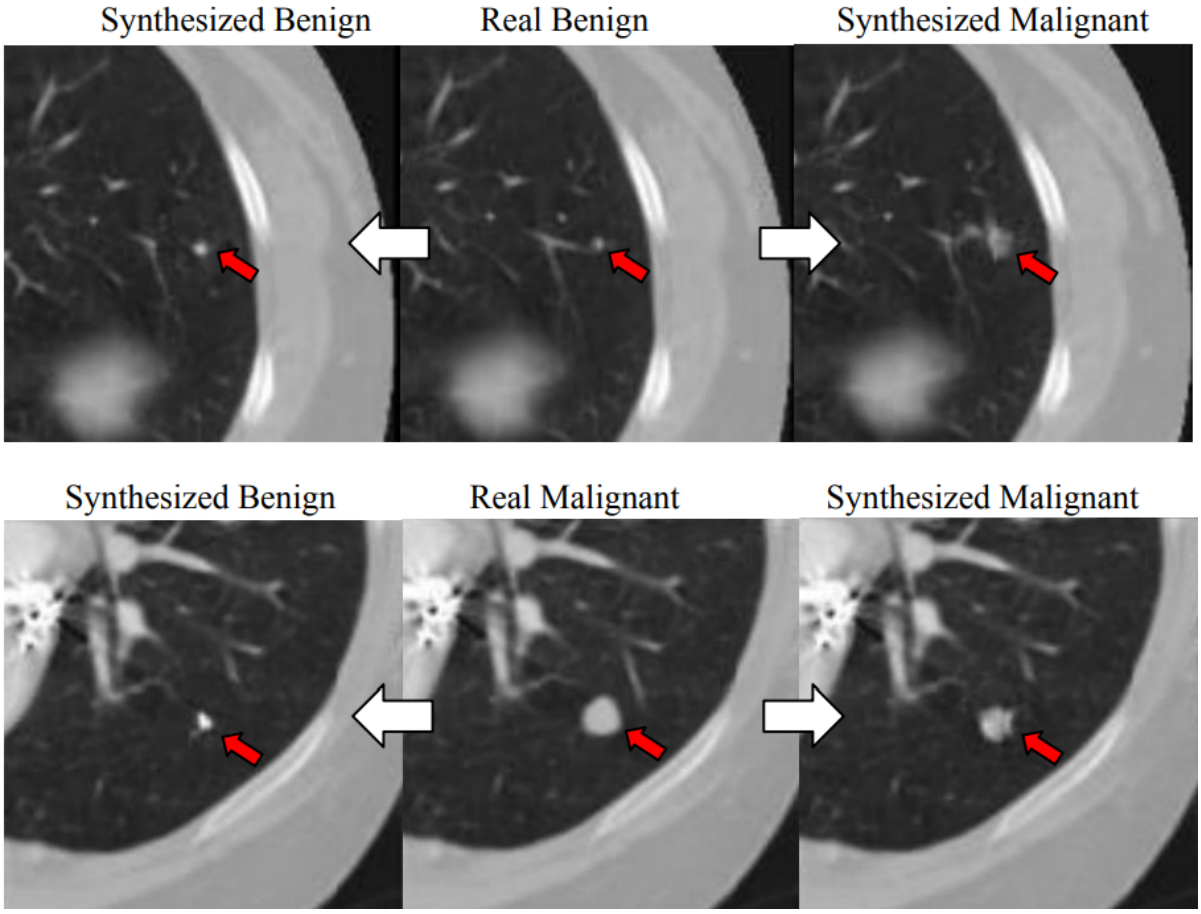


Figure 3.27: Examples of lung nodules (red arrows) generated by the Class-Aware Adversarial model [48].

In another attempt to synthesize lung nodules, in [15] was proposed a 3D Multi-Conditional GAN, capable of generating realistic lung nodules at the desired position, size and attenuation. The generator receives as input a 64^3 volume with the center 32^3 voxels being transformed into noise (where the nodule is located) and is concatenated with a $64^3 \times 6$ volume that represents the conditions (Figure 3.28). The 3D U-Net [50] generator then outputs a 32^3 synthesized nodule that is used in two discriminators, the content discriminator and the nodule discriminator. The context discriminator evaluates whether the real volume (nodule with surroundings) and the generated volume (synthesized nodule with surroundings) are a fake pair. In the end, the nodule discriminator tries to classify real *vs* synthetic nodules with the size and attenuation conditions.

As for evaluation, the metric used was the Competition Performance Metric (CPM) score, which is the average sensitivity of a classifier at different false positive rates, for which a 0.518 score was obtained using the real dataset and 0.550 for the augmented dataset. Having achieved

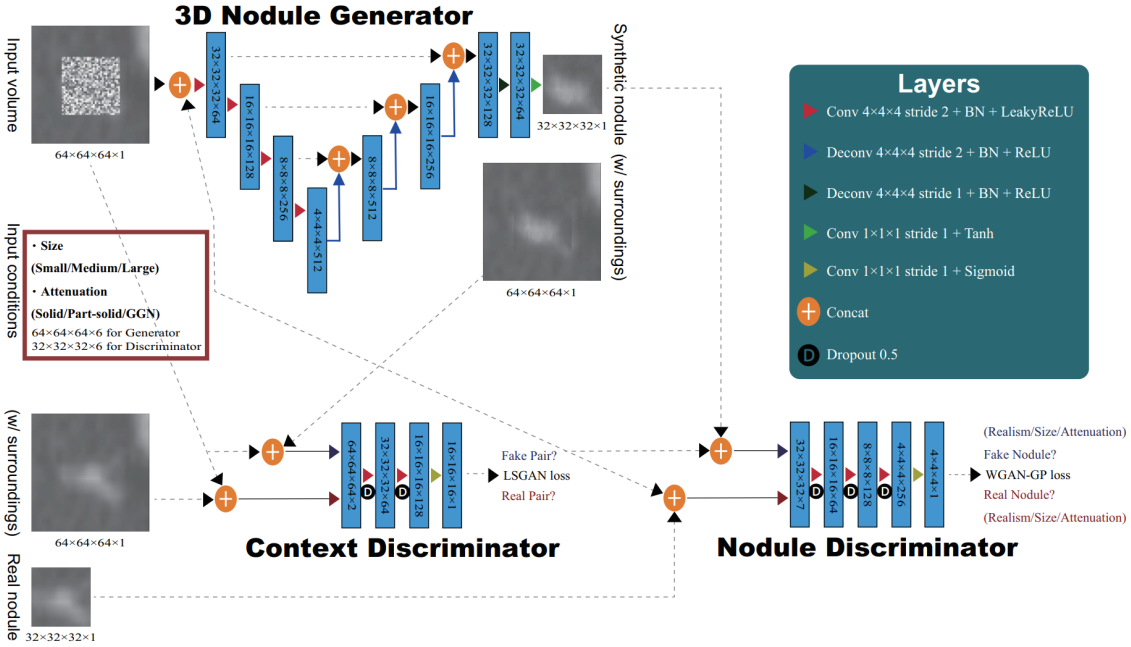


Figure 3.28: 3D MCGAN architecture [15]. The context discriminator and the nodule discriminator learn to classify real and synthesized full volumes and lung nodules, respectively.

perceptually decent results (Figure 3.29) a Visual Turing Test (VTT) (Section 3.6.6) was also performed, where the two physicians obtained an average of 55% and 60.3% accuracy.

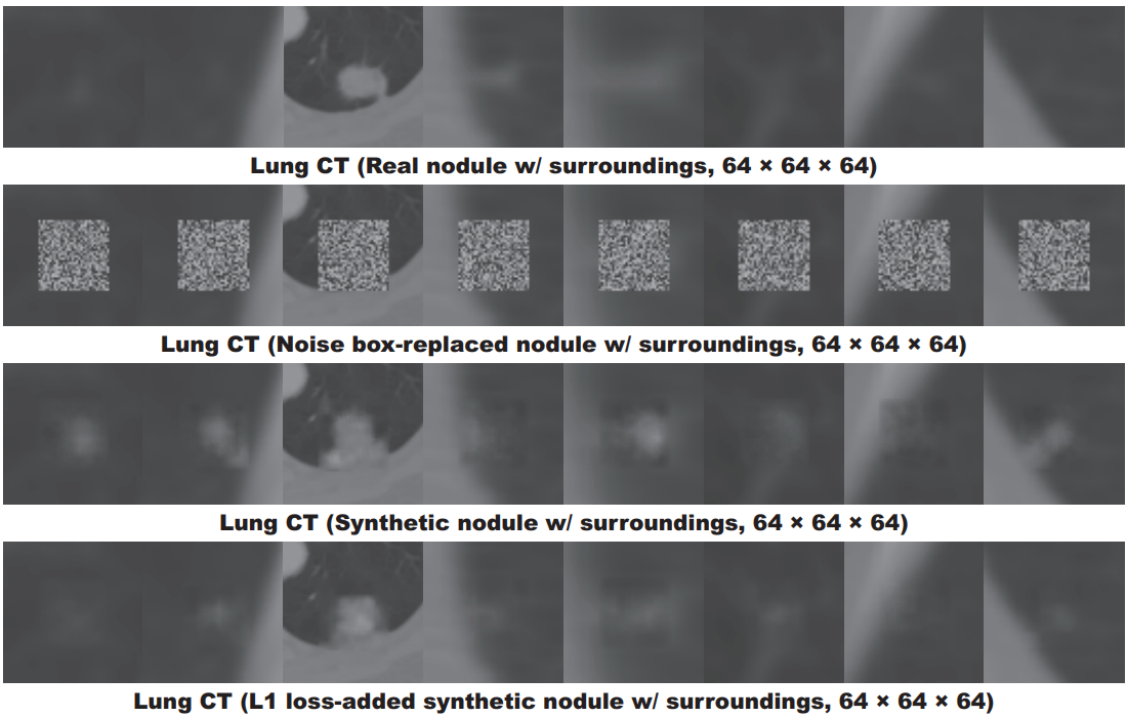


Figure 3.29: A 2D axial view of 3D lung nodules generated by the 3D MCGAN [15].

3.6 Evaluation Metrics

Evaluating GANs is an open problem since there exist several methods but none can be identified as the golden standard. These can be divided into two categories: perceptual studies, which require human observers to analyze and identify synthesized and real samples; objective metrics, which contain similarity scores and distances. This section approaches the most common metrics, as well as, a perceptual study.

3.6.1 Inception Score

Proposed in [51], the IS applies the Inception Model [52] to every generated sample to get the conditional label distribution $p(y|x)$. Samples with meaningful objects ought to have this label distribution with low entropy. As the model is expected to produce diverse outputs, the marginal $\int p(y|x = G(z))dz$ should have high entropy. As a combination of these two, the IS is:

$$exp(\mathbb{E}_x KL(p(y|x)\|p(y))) \tag{3.16}$$

where higher values indicate higher quality and diverse samples.

3.6.2 Mode Score

The Mode Score (MS) [53] is an improvement on the IS, where the MS measures the dissimilarity between the real and the generated distribution. It is given by:

$$exp(\mathbb{E}_x KL(p(y|x)\|p(y)) - KL(p(y)\|p(y*))) \tag{3.17}$$

3.6.3 Fréchet Inception Distance

The FID [54], like the MS, is an improvement on the IS. It is calculated by measuring the Fréchet distance between two multivariate Gaussians with mean and co-variance ($\boldsymbol{m}$, $\boldsymbol{C}$) for the model samples distribution and ($\boldsymbol{m}_w$, $\boldsymbol{C}_w$) for the real dataset distribution. It is given by:

$$\|\boldsymbol{m} - \boldsymbol{m}_w\|_2^2 + Tr(\boldsymbol{C} + \boldsymbol{C}_w - 2(\boldsymbol{C}\boldsymbol{C}_w)^{1/2}) \tag{3.18}$$

3.6.4 Structural Similarity

The Structural Similarity (SSIM) [55] is a method that measures the similarity between two images by using three different components of each one: luminance (l), contrast (c) and structure (s). The SSIM for two images, x and y, is given by:

$$[l(x,y)]^\alpha \cdot [c(x,y)]^\beta \cdot [s(x,y)]^\gamma \tag{3.19}$$

where the exponents α, β and γ are parameters used to adjust the importance of each component and,

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{3.20}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{3.21}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x + \sigma_y + C_3} \tag{3.22}$$

where μ_x and μ_y are the means, σ_x and σ_y are standard deviations, σ_{xy} is the correlation coefficient and C_1,C_2 and C_3 are constants.

3.6.5 Multi-Scale Structural Similarity

The MS-SSIM [56] improves on the SSIM metric by evaluating image details at different resolutions, instead of one. The metric consists in taking as input the two images, interactively applying a low-pass filter and then downsampling the input by a factor of 2. Like the SSIM, the MS-SSIM uses the luminance (l), contrast (c) and structure (s) of the images, however, the luminance is only computed after M - 1 iterations, being that M is the highest scale and M = 1 is the original image. The final calculation is given by:

$$[l_M(x,y)]^{\alpha_M} \cdot \prod_{j=1}^{M} [c_j(x,y)]^{\beta_j} [s_j(x,y)]^{\gamma_j} \tag{3.23}$$

where the exponents α, β and γ are used to adjust the importance of the components.

3.6.6 Visual Turing Test

The VTT [57] consists in asking humans binary questions in order to assess the system's ability to recognize objects and identify attributes and relationships in images. In the context of this document, if said humans are incapable of discriminating between the system's outputs and reality, then the test is successful.

3.7 Summary

In this chapter, the three main types of generative models were explored. Autoregressive models can produce high-quality images, however, they are inefficient to train and are mainly used for image completion or frame interpolation. VAEs are efficient to train and sample from, but output blurry images and thus, not realistic. GANs currently produce the higher quality results of all models, and due to their success are also the most explored framework. However, the biggest disadvantages of GANs are the difficulty of training and the lack of a standard and reliable evaluation metrics, which can lead to subjective visual evaluation.

In medical imaging, existing image-to-image models do not generate full volumes, instead generating only lung nodules. Although the generated samples look realistic, by only generating lung nodules, other conditions that can lead to lung cancer, like emphysema, are ignored. By contrast, noise-to-image models generate the whole 3D scans, with architectures like the 3D-PGGAN being capable of synthesizing realistic samples, which is more in line with the goal of the project. Between all the analyzed medical studies, the resolution was a common aspect with it being 64^3, leading to believe that limitations can be found when trying to improve it.

Many of the metrics used to evaluate image generation models and their outputs are solely focused on 2D data and cannot be adapted to the 3D space, as is the case with the Inception Model based scores. However, metrics that evaluate the data directly, like the SSIM and the MS-SSIM, can be modified to take volumes as input.

Chapter 4

Lung Synthesis

This chapter details the development of the deep learning model used for the synthesis of lung Computed Tomography (CT) scans. To start is introduced the dataset that is used to train the model, as well as its pre-processing steps. This is followed by a description of the model, its structure and details regarding its training procedure.

4.1 LIDC-IDRI

Datasets are a key component of deep learning and when implementing a generative model, and not only high-quality data is needed, but a dataset that is frequently used in other studies for comparison, is also valuable. A qualifier for this task is the Lung Image Database Consortium (LIDC) [21], a publicly available dataset that consists of 1018 diagnostic and lung cancer screening thoracic CT scans (Figure 4.1), where each sample has an associated XML file containing the results of an annotation process performed by four radiologists to identify lesions. This process occurred in two phases: the initial blind-read phase, where each radiologist independently reviewed each scan and marked lesions; and the unblinded-read phase, where the radiologists independently reviewed the scan with the anonymized marks of all radiologists to form a final opinion. A summary of the lesions identified by radiologists is as follows:

- 7371 lesions marked as a nodule by at least one radiologist;
- 2669 were marked as a nodule $\geq$ 3mm by at least one radiologist;
- 928 were marked as a nodule $\geq$ 3mm by all four radiologists.

The reviews were executed independently and anonymously as a way to identify all possible lesions in every sample without requiring forced consensus between professionals.

Figure 4.1: A CT slice sample from the LIDC dataset [21].

4.1.1 Pre-Processing

Thoracic CT scans contain a lot of data aside from just the lungs, like muscles and bones. As this extra data makes a great part of the scan, synthesizing all that information would increase the difficulty of the model training process. To avoid this, the data pre-processing consisted in segmenting the lungs from the CT images, which was based on an existing process that can be found on Kaggle's Data Science Bowl 2017[1]. In the first step, the scans are normalized between the minimum and maximum Hounsfield units (HU) values of -1000 and 400, respectively. These values are used so that the empty space outside the scan is treated as air inside the scan, singling out the patient in the image (Figure 4.2).

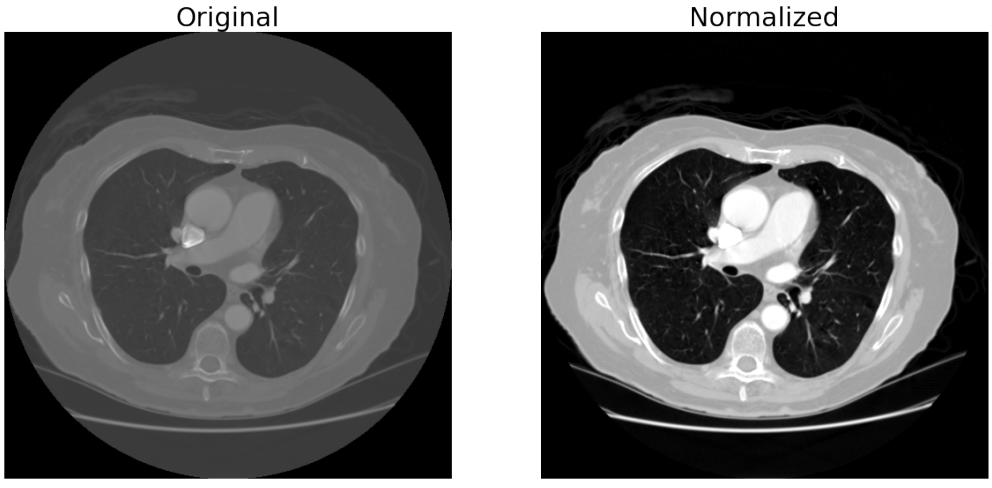


Figure 4.2: An original and a normalized CT scan slice. The normalization is used as a way to turn air and space outside of a scan into the same values.

[1]Data Science Bowl 2017, 19-09-2021

Afterward, a mask for the lung is created using a labeling process. Firstly, the scan is transformed into a binary volume using an empirically found threshold of 0.4, resulting in air being valued at 0 and the rest of the scan at 1 (Figure 4.3 a). This array is then labeled into different regions by accessing which of the voxel neighbors have the same value as itself with a maximum distance of 2 orthogonal hops. With this, the higher left most voxel label is picked, assuming that it represents empty space (value 0), and the voxels corresponding to that label are given the value of 1, the same value as the area around the lung (Figure 4.3 b). At this point, the structures inside the lung (i.e. bronchi) are filtered by labeling the volume slice by slice and turning the regions valued at 1 that are not the biggest (i.e. the regions inside the lung), into a value of 0 (Figure 4.3 c). To finalize the mask generation, the binary values are inverted so that 0 refers to air and 1 refers to lung and the array is put through a dilation that fills possible existing holes inside the lungs (Figure 4.3 d). Additionally, for a scan to contain as much lung and as little air as possible, the volumes were cropped on every axis until the binary mask was reached, with a margin of 5 voxels (Figure 4.3 e). Finally, the mask is used on top of the original scan, revealing the segmented lung (Figure 4.3 f).

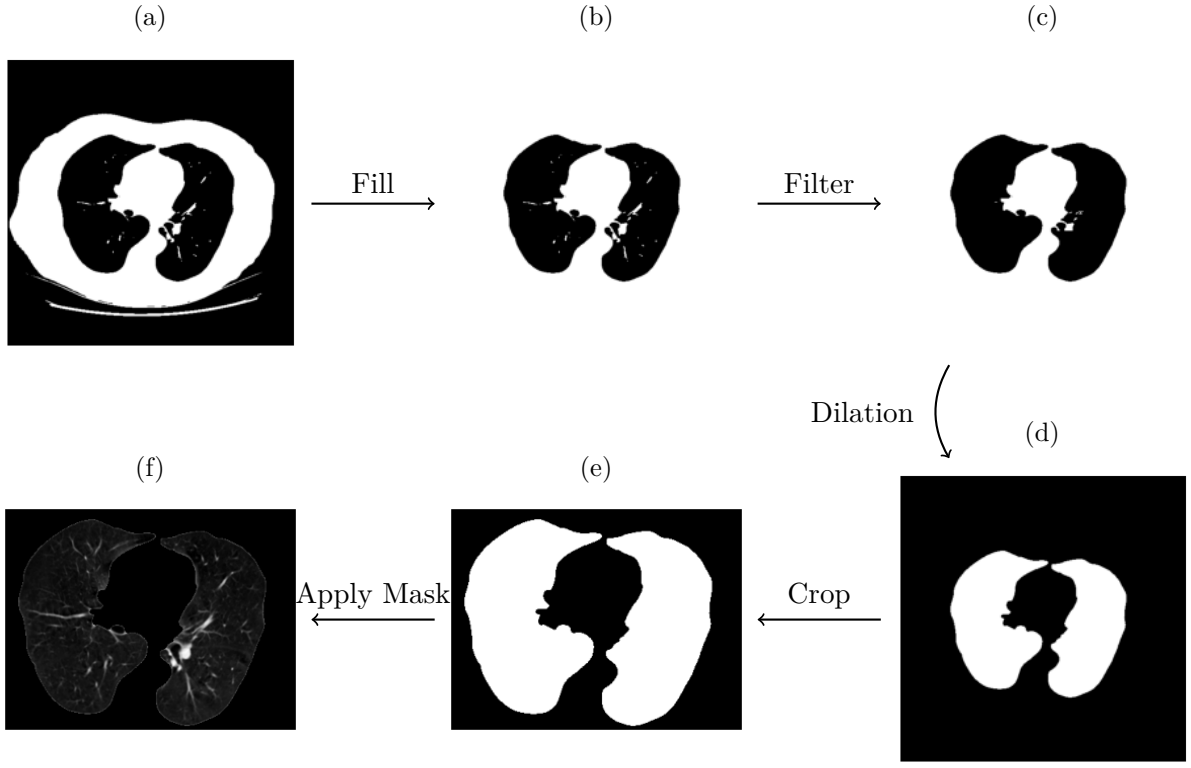


Figure 4.3: Lung segmentation process. First, the volume is transformed into a binary array (a) and the air around the person is transformed into the same value as the region around the lung (b). The structures inside the lung are then filtered from the scan (c) and the array goes through a dilation to fill potential holes in the mask (d). Afterward, the mask is cropped to fit the lung (e) and the mask is applied to the real data (f).

In CT scans, different volumes may have different spacing between pixels and slices, putting them in a different space. As such, the volumes were resized so that each pixel represents 1mm.

In order to minimize the number of resizes needed (avoiding quality loss), this step was executed along with a resize to a 256^3 resolution. The resize factor for this transformation is calculated by dividing 256 by the largest length axis out of all the samples, however, not every scan available was used. As shown in Figure 4.4, the scans consisted of two groups of maximum axial size having a mean of 359, standard deviation of 112 but a maximum of 831. This variation derives from samples where segmentation was not successful, having failed because for example the original scan is cropped to the lung and there is no empty space, making the mask generation fail or during the cropping step, some "lung" was found before the actual lung, making the final volume much larger than it should be. Therefore, to avoid having to resize volumes to too small of a size, the samples used were the ones found in the same group as the mean, having the lowest and highest axial size of 235 and 418 respectively, totaling 717 samples. As such, at the end of the pre-processing, each pixel did not represent 1 mm, but 0.61 mm. After resizing, the scans were padded with zeros to become 256^3 of resolution having the lung in the center (Figure 4.5).

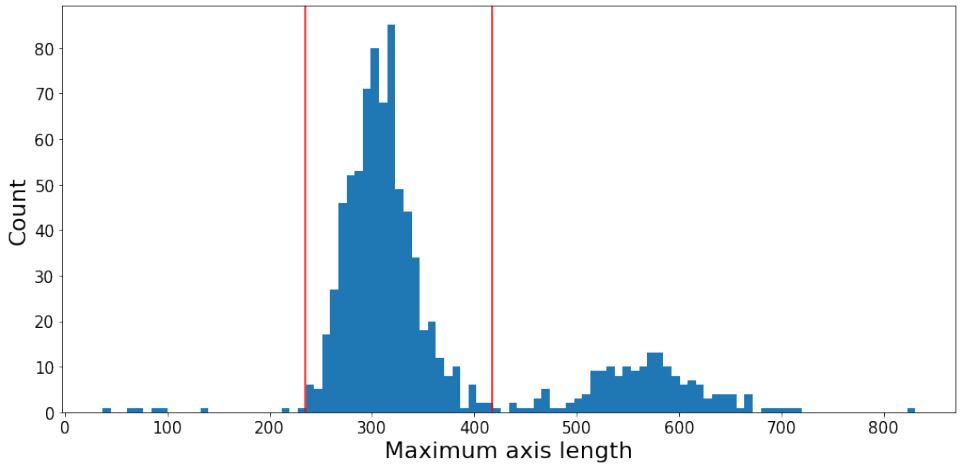


Figure 4.4: Histogram of the biggest axis for each scan. The red vertical lines indicate the lowest and highest values of the samples included in the final dataset.

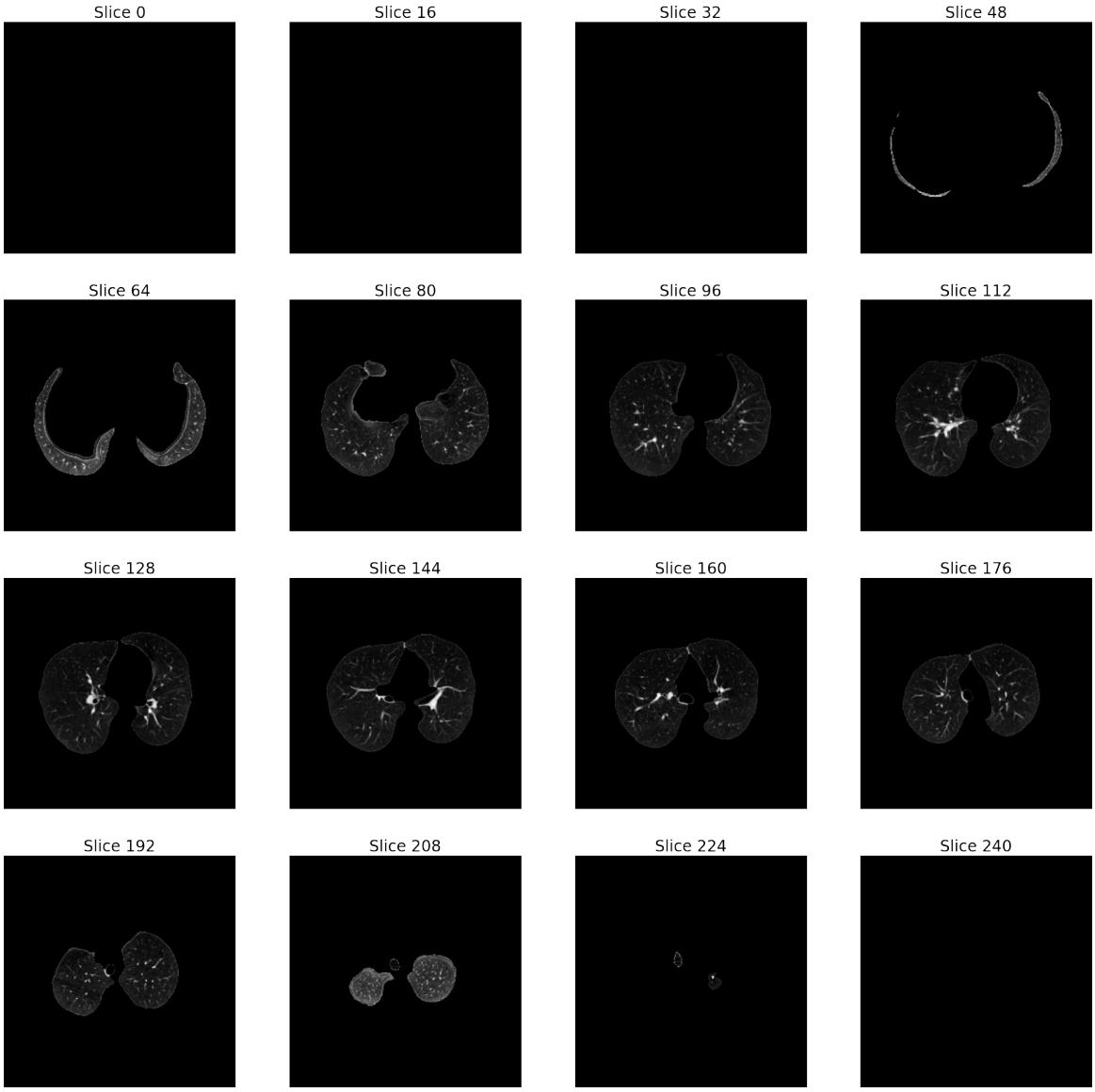


Figure 4.5: Example of a full pre-processed sample, at various slices.

4.2 3D Progressive Growing GAN

The 3D Progressive Growing GAN (PGGAN) used was based on the implementation of Anders Eklund [17], which itself was built on top of the original PGGAN [41] by adapting the architecture to the 3D space, where an extra dimension was added to relevant code calls and 2D convolutions were changed into 3D convolutions.

4.2.1 Structure

The network consists of a group of blocks with one for each resolution, being comprised of an upsampling or downsampling operation followed by two convolutions with the exception of the

first and last blocks for the generator and discriminator, that begin and end with fully connected layers, respectively (Figure 4.6).

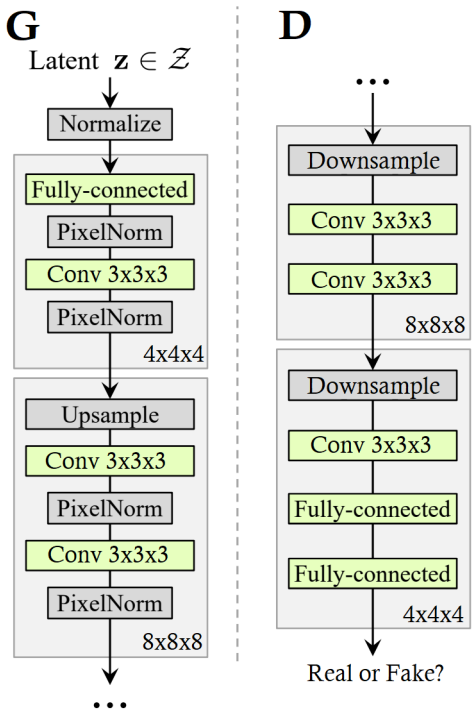


Figure 4.6: The 3D PGGAN block structure of the generator (left) and discriminator (right) (adapted from [43]).

While the generator blocks do not make use of batch, layer or weight normalization, a pixel-wise normalization is applied on features vectors after each convolutional layer, being is calculated as:

$$b_{x,y,z} = \frac{a_{x,y,z}}{\sqrt{\frac{1}{N}\Sigma_{j=0}^{N-1}(a^j_{x,y,z})^2 + \epsilon}} \tag{4.1}$$

where $\epsilon = 10^{-8}$ (to avoid division by zero), N is the number of feature maps and $a_{x,y,z}$ and $b_{x,y,z}$ are the original and normalized feature vector in voxel (x, y, z), respectively.

The latent vector size was set at 128 and all convolutions make use of 128 filters with the exception of the highest resolution layers, where 64 filters were used due to memory constraints. The convolutions kernel size was set at 3 with a stride of 1, being now volumetric, and making use of the LeakyReLU activation function with a slope of 0.2.

Although the network contains 3D convolutions, the final model totaled approximately 10.7 million trainable parameters with the generator and discriminator having 5.3 million each one, quite a bit lower than the example given by Karras et. al. [41] where their model totaled more than 45 million. This occurs due to two reasons:

- Unnecessary high number of filters in the lower resolutions: the 2D version model actually contained most of its parameters at the lower resolutions, leading to less impact in the final outputs as shown later by the authors [44];

- Memory limitations: the model was trained by fully utilizing two 11GB GPUs, while in the 2D original version the authors recommend an 8×16GB GPU setup and the 3D original version a 4×32GB GPU configuration, with the latter maxing out at a resolution of 64^3.

4.2.2 Training

The training process consists of both the generator and discriminator starting by generating and evaluating 4^3 volumes respectively, with the level of detail being doubled throughout training until a 128^3 resolution is reached. When the resolution is doubled, the new layers are added smoothly by treating them as a residual block, whose weight goes from 0 to 1 linearly with training (Figure 4.7).

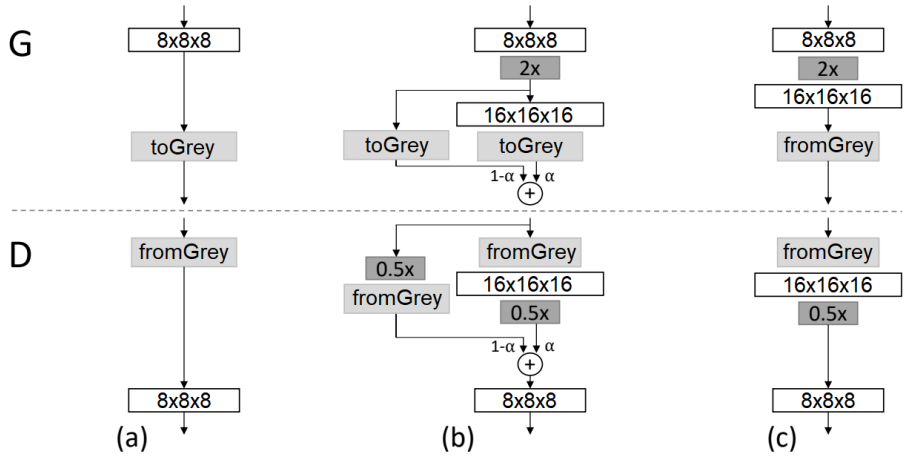


Figure 4.7: Process used by the 3D PGGAN to transition between resolutions, in this case from 8^3 (a) to 16^3 (c). During the transition (b), the higher resolution layers are treated as a residual block, having their weight α increase from 0 to 1 which is used to perform a weighted sum between inputs. The 2× and 0.5× refer to doubling and halving the volume resolution to using nearest neighbor filtering and average pooling, respectively (adapted from [41]).

To train the model, the discriminator and the generator are trained intermittently with one minibatch each, with the total procedure being defined by how many real samples the discriminator has evaluated. For each resolution, the discriminator was passed through 300k images in both normal training and during resolution transitions, with the last resolution being trained for an additional 300k for stabilization. Unlike in the original 3D implementation [17], this model reaches a maximum resolution of 128^3 and it was found that as resolution increased and minibatch size decreased, higher learning rates compromised training stability, therefore the learning rate was maintained or decreased over the resolutions (Table 4.1).

In the resolutions 64^3 and 128^3 the learning rates were not static due to instability while transitioning between the previous resolution and themselves. The last two resolutions consisted in a progressive increase of learning rate because during transitions was when the discriminator found itself most vulnerable since the data it was fed was constantly changing, the minibatch size was considerably low and restarting the training process would be very time consuming.

	Resolution					
	4^3	8^3	16^3	32^3	64^3	128^3
LR G	0.0004	0.0004	0.0004	0.0004	0.0001-0.0002	0.00005-0.00012
LR D	0.0005	0.0005	0.0005	0.0005	0.0003-0.0005	0.0002-0.0004
Minibatch size	128	64	32	16	4	2
Train Time	5m	20m	1h 20m	6h 50m	2d 8h 5m	17d 2h 20m

Table 4.1: Learning rates, minibatch size and training time of the 3D PGGAN for each resolution. The model was trained using two NVIDIA RTX 2080 Ti having 11GB of memory each.

Additionally, the generator would often learn about flaws in the discriminator which would lead to mode collapse (Figure 4.8) and thus, its learning rate had to be kept lower than its opponent.

Another occurrence of this flaw hunting made by the generator manifested in the first and last slices of its outputs which are always mostly grey when in the real samples are black (Figure 4.9). Most likely this happens because in the early stages of training, when resolution is low, the real data has values other than zeros (lung) in the outer slices and the generator learns that structure but sticks to it, while the discriminator doesn't penalize that behavior. When the resolution increases, since the lungs themselves are not found in the borders of the scan, the discriminator ignores the grey outer slices and focuses solely on the lungs, leaving the situation unresolved throughout the whole training process.

4.3 Summary

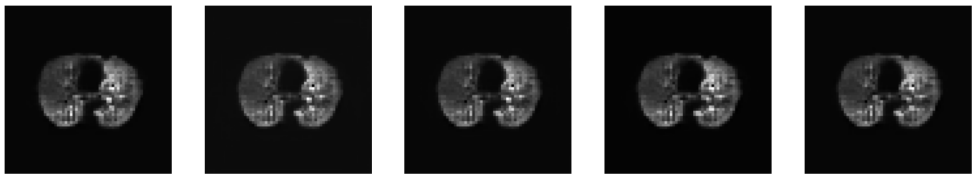
Although the LIDC dataset has the upside of being publicly available, the amount of samples it contains is not as high as it should be considering the context of this work, a deep learning application. This situation is emphasized even more after the lung segmentation, as the process is not flawless and samples had to be discarded. Having only 717 examples in the final dataset could mean that the model was not only limited by computational resources but also in the diversity of the data it trained on.

The progressive architecture of the 3D PGGAN guides the model to quickly figure out the structure of the data it is supposed to synthesize, with the generator outputting visually similar results to the real data very early on. However, the architecture has its drawbacks like the hyperparameter search, where optimal learning rates and minibatch sizes have to be found for every resolution and even for both the generator and discriminator, making that task alone require a great amount of time. Additionally, its greatest strength may also be its greatest weakness as the progressive growth can result in artifacts that are not dealt with at the beginning of training, remain unscathed and have a significant impact on the final outputs of the model.

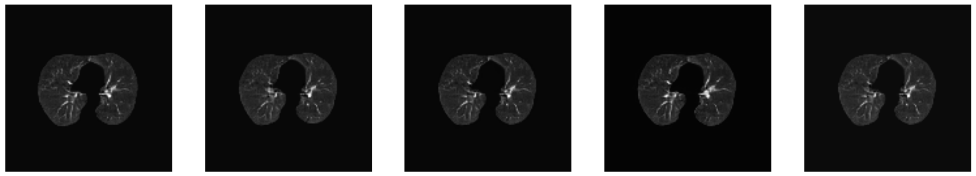
(a) 8^3

(b) 16^3

(c) 32^3



(d) 64^3



(e) 128^3

Figure 4.8: Mode collapse in all resolutions. Note that for any given resolution, each image represents a unique latent code while the result is the same, meaning that the generator has found a flaw in the discriminator and capitalized on it.

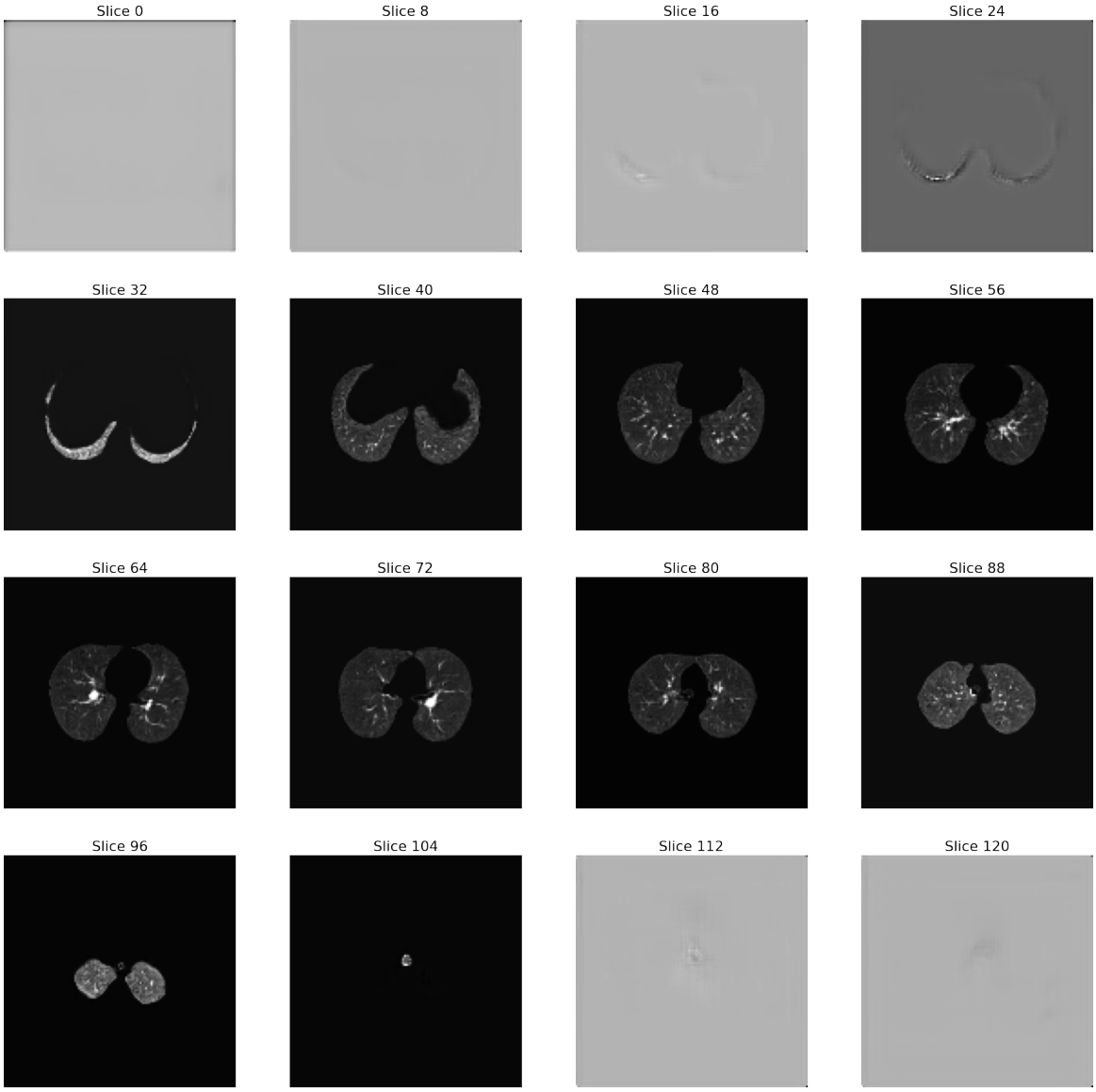


Figure 4.9: Output example of the 3D PGGAN. In all of the produced samples, the first and last slices were mostly grey.

Chapter 5

Results

The lack of easily interpretable metrics to evaluate synthesized samples of generative models has proven to be a difficult task and this problem is only emphasized in the 3D space. In this chapter, the model is first evaluated by comparing the structural similarity between samples, with both real and generated sets of data. This is followed by the detailing and analysis of the Visual Turing Test (VTT) performed.

5.1 3D MS-SSIM

While the Multi-Scale Structural Similarity (MS-SSIM) metric does not assess the model output quality or similarity in relation to the original data, it evaluates the diversity of a given sample set. This is important to evaluate Generative Adversarial Networks (GANs) as one of their common problems is mode collapse, which happens when the generator learns to reproduce only a few or even just one output. If this occurs, the MS-SSIM score will be much higher for the generated samples than for the real ones.

The 3D MS-SSIM was calculated by comparing 10K volumes, for both (pre-processed) real and generated samples (Table 5.1). Additionally, to have a better sense of the scores obtained, the MS-SSIM was also computed for a network state in which mode collapse had occurred.

Data	**Score**	**Std**
Real	0.848	0.031
Final Model	0.788	0.050
Mode Collapse	0.973	0.008

Table 5.1: 3D MS-SSIM results.

Although the scores seem high, especially compared to 2D studies like the 2D PGGAN where the MS-SSIM was 0.284, there is the factor of the data itself. Both real and generated samples contain a lot of empty space around the lung and this is further emphasized by the 3D aspect of

the scans, making by default the MS-SSIM seem high.

Having a higher score in the real samples means that the diversity of the dataset used is low, however, this confirms that the latent space was not limiting the model capabilities and so, there was no need for a higher latent space vector size. With a better score for the generated samples, it is reasonable to assume that sample diversity is better than the real data, however, that is not entirely true. In this case, the better score can be at least partially attributed to the small subset of generated examples that are not realistic (Figure 5.1), which happened around in one out of five cases. These unrealistic samples also give sense to the higher value in the standard deviation, proving that the generated set contains bigger inconsistencies.

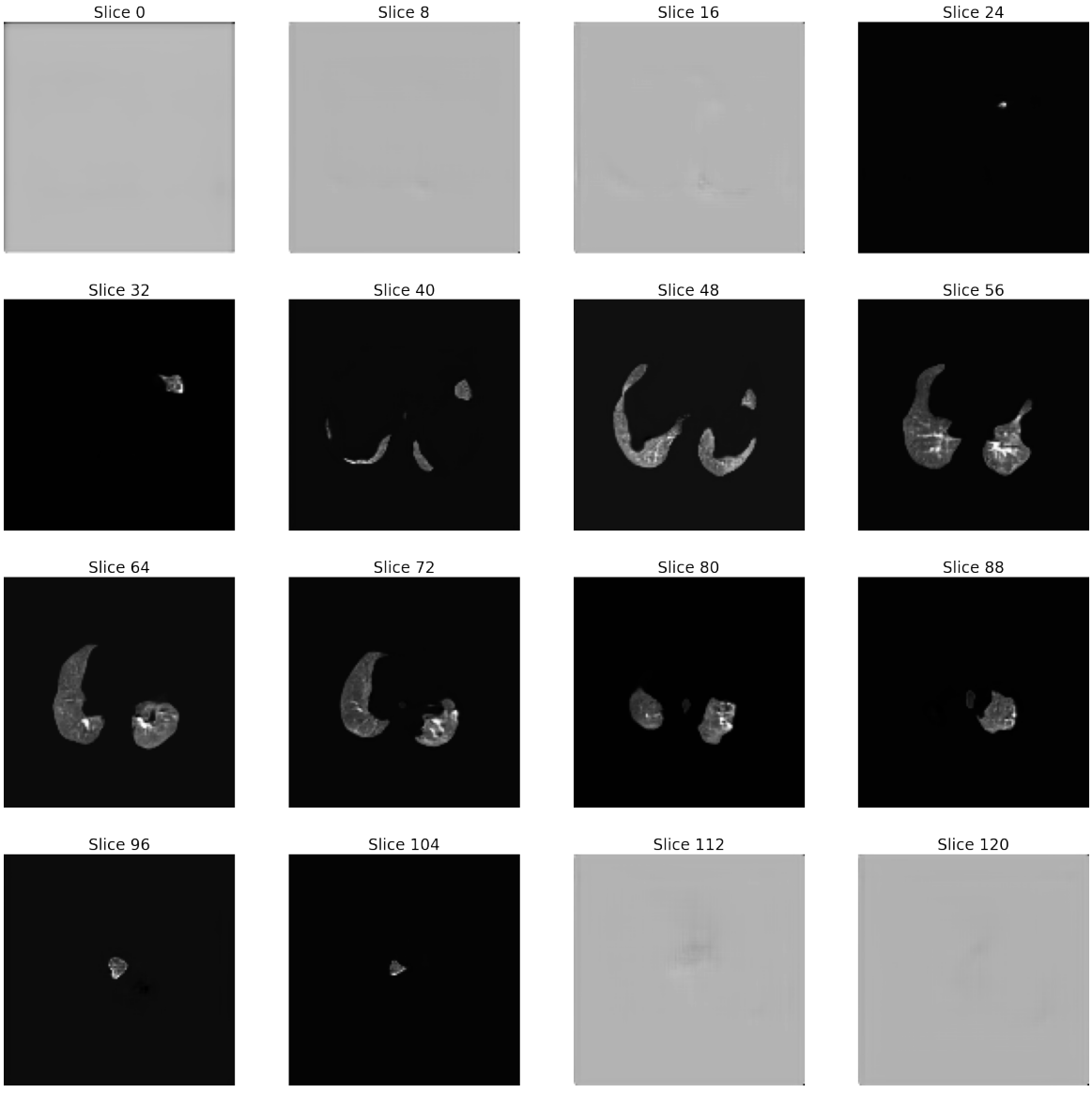


Figure 5.1: An unrealistic sample produced by the 3D PGGAN. The lower MSSIM score for the generated sample set can be attributed to the model outputs that are not realistic.

5.2 Visual Turing Test

The VTT test was performed by asking two radiologists interns (R1 and R2) to evaluate 50 unique lung CT scans as real or generated. The test set consisted of 25 real and 25 synthesized scans, although the subjects were not given this information. The real scans were randomly selected from the pre-processed dataset and downsampled to 128^3 to match the synthesized examples, while the generated scans were manually selected from a 200 sample set of outputs from the 3D Progressive Growing GAN (PGGAN). As the grey slices in the generated samples would be an obvious sign of synthesis, slices were removed from the top and bottom for the scans to contain lung exclusively, including in the real examples. For the calculations of the confusion matrices, the real scans were considered as negative and the generated as positive (Tables 5.2 and 5.3). Furthermore, the matrices were then used to calculate 5 metrics (Table 5.4): accuracy, precision, recall, false omission rate (FOR) and negative predicted value (NPV).

		Predicted	
		Positive	Negative
Actual	Positive	TP = 24	FN = 1
	Negative	FP = 5	TN = 20

Table 5.2: Confusion Matrix of R1.

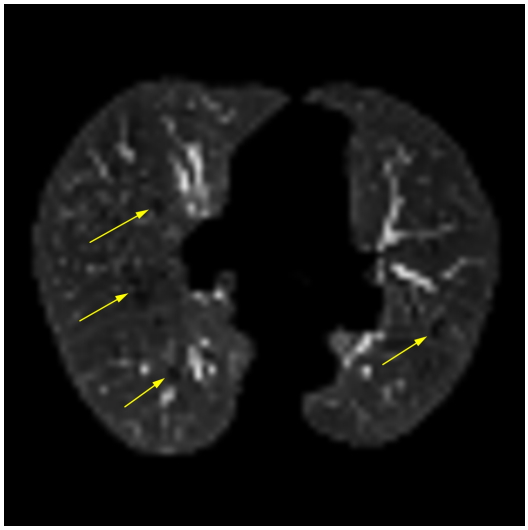
		Predicted	
		Positive	Negative
Actual	Positive	TP = 9	FN = 16
	Negative	FP = 18	TN = 7

Table 5.3: Confusion Matrix of R2.

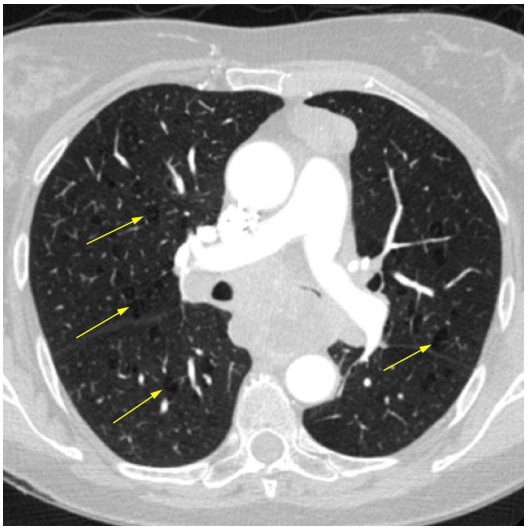
	Accuracy	Precision	Recall	FOR	NPV
R1	0.88	0.83	0.96	0.05	0.95
R2	0.32	0.33	0.36	0.70	0.30

Table 5.4: Evaluation metrics for both of the radiologists' VTT.

Being a VTT, the ideal results would consist of an accuracy of 0.5, precision of 0, a recall of 0, a FOR of 0.5 and an NPV of 0.5, or in other words, predictions would only be negative. Judging the results, the FOR suggests that the generated samples were far more convincing to R2 than to R1. However, the radiologists also presented difficulty in evaluating real samples as demonstrated by the much lower NPV by R2 and with the false positives (FP) being higher than the false negatives (FN) in both cases. This leads to believe that the discrepancies between evaluations come from the subjects' experience of analyzing Computed Tomography (CT) scans with higher resolution and greater detail than the ones presented in this experiment. To further this hypothesis, after classification, the radiologists were asked to give details regarding their decisions on some of the samples. The radiologist R1 mentioned that in one of the scans that was classified as generated, it seemed as some spots lacked information (Figure 5.2 a). However, the sample was real (Figure 5.2 b) and those same spots were actually a sign of emphysema. This means that in lower resolutions, features that would normally indicate the presence of pathologies can be regarded as artifacts or errors in the scans, leading to misclassification.



(a) Annotated slice by R1 (cropped from the resolution 128^2).



(b) Example of feedback by radiologist R1 (cropped from the resolution 512^2).

Figure 5.2: Feedback example given by R1. Image (a) is annotated with arrows pointing to spots which lead the radiologist to classify the scan as generated. Image (b) is annotated with arrows pointing to the same spots, in the real scan. The lower resolution made the annotated spots seem like errors when they would normally indicate emphysema.

Regarding the correctly classified generated samples, radiologist R1 pointed out that a decisive factor was often the oversized airways of the bronchi in relation to the actual size of the CT itself (Figure 5.3).

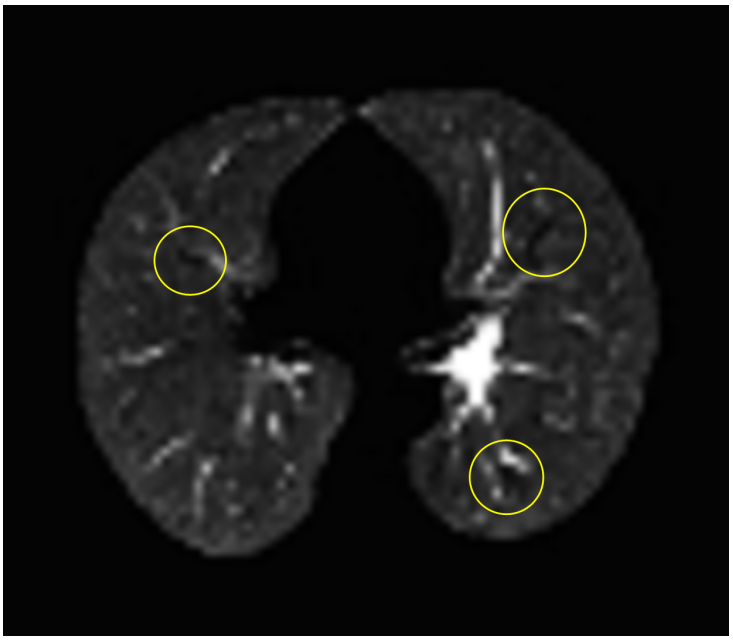


Figure 5.3: Generated sample with annotated oversized bronchi airways. The size of these structures was often a decisive factor in distinguishing examples.

Overall, both radiologists reported that the resolution made it difficult to classify the scans, having to sometimes resort to intuition instead of using objective analysis. As information to assess the veracity of examples was lacking, there is room to suspect whether the results of the evaluations were either luck or misfortune.

5.3 Summary

The evaluation of this works model exemplifies how difficult it can be to evaluate generative models. While the 3D MS-SSIM and the VTT were analyzed, it is hard to come to a conclusion regarding the diversity and quality of the synthesized samples. The MS-SSIM proves that the network did not suffer from mode collapse and although the score was better for generated than for real samples, this can be attributed to unrealistic outputs. In the VTT the two radiologists achieved almost symmetrical results, with both having stated that the 128^3 resolution was a clear limitation in their judgment. Thus, it is unclear if the synthesis of the 3D lung CT scans can be considered successful or not.

Chapter 6

Conclusion

This chapter finalizes the work done in this dissertation by summarizing the document, drawing its key conclusions and outlying possible improvements to implement in the future.

6.1 Summary

The purpose of this work was to create a model capable of synthesizing 3D lung Computed Tomography (CT) scans with realistic and spatially coherent characteristics, having medical professionals unable to distinguish them from real scans. To accomplish this, the lung anatomy and most common pathologies were studied as well as the technology behind the CT imaging technique. Afterward, the literature of generative models was reviewed, being mainly focused on Generative Adversarial Networks (GANs) as not only they have been the most explored in regular 2D applications, they also dominate the biomedical 3D space.

The Progressive Growing GAN (PGGAN) architecture was made with training stability in mind and had already been successfully used to synthesize 3D data. This work adapts that same structure to generate volumetric lung scans while using the publicly available LIDC dataset to train the model. As 3D data contains exponentially more information than 2D images, the 3D-PGGAN had to be limited to a resolution of 128^3 and was trained for almost 20 days. Additionally, the hyper-parameter search turned out to be more difficult than expected, with it having to be done for each resolution.

The model was capable of synthesizing spatially coherent scans with better diversity than the training dataset, having however some unrealistic scans in between. Furthermore, a Visual Turing Test (VTT) was performed by two radiologists interns, where the subjects had to classify 50 scans as being real or generated. Mainly due to the limited resolution, the results turned out to be inconclusive as they were symmetrical between the radiologists and with the better performer misconceiving more real samples than synthesized.

Changing into the 3D space brings great potential to the generation of the CT lung scans field, by

enabling the analysis of the data as it is done by radiologists in a medical environment. However, this shift comes at a great cost of resolution, with it being the main drawback of this work.

6.2 Future Work

As the main limitation of this work is the resolution of the generated scans, the greatest and highest priority improvement is to increase the resolution to at least 256^3. However, here are listed other suggestions to improve the current state of the work:

- Use a bigger dataset with higher data diversity to reduce similarity in generated samples;
- Use labels to condition the model, enabling the creation of purposefully balanced datasets;
- Evaluate the model by comparing results of a trained deep learning classifier with and without generated samples;
- Integrate more subjects in the Visual Turing Test to avoid inconclusive results.

Bibliography

[1] Wild, C.P., Weiderpass, E. and Stewart, B.W. *Cancer research for cancer prevention World Cancer Report*. 2020. ISBN: 9789283204473.

[2] Lu, T., Yang, X., Huang, Y. et al. Trends in the incidence, treatment, and survival of patients with lung cancer in the last four decades. *Cancer Management and Research*, Volume 11:943–953, 1 2019. ISSN: 1179-1322. doi:10.2147/CMAR.S187317.

[3] Lung nodule and cancer detection in computed tomography screening. volume 30, pages 130–138. Lippincott Williams and Wilkins, 3 2015. doi:10.1097/RTI.0000000000000140.

[4] Hussein, S., Cao, K., Song, Q. and Bagci, U. Risk stratification of lung nodules using 3d cnn-based multi-task learning. *CoRR*, abs/1704.08797, 2017.

[5] Tanaka, F. and Aranha, C. Data augmentation using gans. *CoRR*, abs/1904.09135, 2019.

[6] van den Oord, A., Kalchbrenner, N. and Kavukcuoglu, K. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016.

[7] Kingma, D.P. and Welling, M. Auto-encoding variational bayes, 2014.

[8] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M. et al. Generative adversarial networks, 2014.

[9] Yi, X., Walia, E. and Babyn, P.S. Generative adversarial network in medical imaging: A review. *CoRR*, abs/1809.07294, 2018.

[10] Abramian, D. and Eklund, A. Generating fmri volumes from t1-weighted volumes using 3d cyclegan. *arXiv*, 7 2019.

[11] Huang, Y., Zheng, F., Cong, R. et al. Mcmt-gan: Multi-task coherent modality transferable gan for 3d brain image synthesis. *IEEE Transactions on Image Processing*, 29:8187–8198, 2020. doi:10.1109/TIP.2020.3011557.

[12] Wang, Y., Yu, B., Wang, L. et al. 3d conditional generative adversarial networks for high-quality pet image estimation at low dose. *NeuroImage*, 174:550 – 562, 2018. ISSN: 1053-8119. doi:https://doi.org/10.1016/j.neuroimage.2018.03.045.

[13] Yu, B., Zhou, L., Wang, L. et al. 3d cgan based cross-modality mr image synthesis for brain tumor segmentation. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 626–630, 2018. doi:10.1109/ISBI.2018.8363653.

[14] Jin, D., Xu, Z., Tang, Y. et al. Ct-realistic lung nodule simulation from 3d conditional generative adversarial networks for robust lung segmentation. *CoRR*, abs/1806.04051, 2018.

[15] Han, C., Kitamura, Y., Kudo, A. et al. Synthesizing diverse lung nodules wherever massively: 3d multi-conditional gan-based ct image augmentation for object detection. In *2019 International Conference on 3D Vision (3DV)*, pages 729–737, 2019. doi:10.1109/3DV.2019.00085.

[16] Zhu, J.Y., Park, T., Isola, P. and Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.

[17] Eklund, A. Feeding the zombies: Synthesizing brain volumes using a 3d progressive growing gan, 2020.

[18] McWilliams, A., Tammemagi, M.C., Mayo, J.R. et al. Probability of cancer in pulmonary nodules detected on first screening ct. *New England Journal of Medicine*, 369(10):910–919, 2013. PMID: 24004118. doi:10.1056/NEJMoa1214726.

[19] Wilson, D.O., Weissfeld, J.L., Balkan, A. et al. Association of radiographic emphysema and airflow obstruction with lung cancer. *American Journal of Respiratory and Critical Care Medicine*, 178:738–744, 10 2008. ISSN: 1073449X. doi:10.1164/rccm.200803-435OC.

[20] Moore, KL, Dalley et al. Clinically oriented anatomy, 7th edition, isbn 978-1-4511-1945-9. *Clinical Anatomy*, 27(2):274–274, mar 2014. ISSN: 08973806. doi:10.1002/ca.22316.

[21] Armato, S.G., McLennan, G., Bidaut, L. et al. The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): A completed reference database of lung nodules on CT scans. *Medical Physics*, 38(2):915–931, 2011. ISSN: 00942405. doi:10.1118/1.3528204.

[22] Koshiol, J., Rotunno, M., Consonni, D. et al. Chronic obstructive pulmonary disease and altered risk of lung cancer in a population-based case-control study. *PloS one*, 4(10):e7380, oct 2009. ISSN: 1932-6203 (Electronic). doi:10.1371/journal.pone.0007380.

[23] Li, Y., Swensen, S.J., Karabekmez, L.G. et al. Effect of emphysema on lung cancer risk in smokers: A computed tomography-based assessment. *Cancer Prevention Research*, 4:43–50, 1 2011. ISSN: 19406207. doi:10.1158/1940-6207.CAPR-10-0151.

[24] Hansell, D.M., Bankier, A.A., MacMahon, H. et al. Fleischner society: Glossary of terms for thoracic imaging. *Radiology*, 246:697–722, 3 2008. ISSN: 00338419. doi:10.1148/radiol.2462070712.

[25] Harzheim, D., Eberhardt, R., Hoffmann, H. and Herth, F.J. The solitary pulmonary nodule. *Respiration*, 90:160–172, 8 2015. ISSN: 14230356. doi:10.1159/000430996.

[26] Results of initial low-dose computed tomographic screening for lung cancer. *New England Journal of Medicine*, 368(21):1980–1991, 2013. PMID: 23697514. doi:10.1056/NEJMoa1209120.

[27] Goodfellow, I.J. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017.

[28] Chen, X., Mishra, N., Rohaninejad, M. and Abbeel, P. Pixelsnail: An improved autoregressive generative model. *CoRR*, abs/1712.09763, 2017.

[29] Hinton, G.E. and Zemel, R. Autoencoders, minimum description length and helmholtz free energy. In Cowan, J., Tesauro, G. and Alspector, J., editors, *Advances in Neural Information Processing Systems*, volume 6, pages 3–10. Morgan-Kaufmann, 1994.

[30] Wei, R., Garcia, C., El-Sayed, A. et al. Variations in variational autoencoders - a comparative evaluation. *IEEE Access*, 8:153651–153670, 2020. ISSN: 21693536. doi:10.1109/ACCESS.2020.3018151.

[31] Gui, J., Sun, Z., Wen, Y. et al. A review on generative adversarial networks: Algorithms, theory, and applications, 2020.

[32] Radford, A., Metz, L. and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[33] Mirza, M. and Osindero, S. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[34] Odena, A., Olah, C. and Shlens, J. Conditional image synthesis with auxiliary classifier gans, 2017.

[35] Russakovsky, O., Deng, J., Su, H. et al. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[36] Isola, P., Zhu, J., Zhou, T. and Efros, A.A. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[37] Ronneberger, O., Fischer, P. and Brox, T. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[38] Long, J., Shelhamer, E. and Darrell, T. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014.

[39] Cordts, M., Omran, M., Ramos, S. et al. The cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685, 2016.

[40] Rusu, A.A., Rabinowitz, N.C., Desjardins, G. et al. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

[41] Karras, T., Aila, T., Laine, S. and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[42] Gulrajani, I., Ahmed, F., Arjovsky, M. et al. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[43] Karras, T., Laine, S. and Aila, T. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018.

[44] Karras, T., Laine, S., Aittala, M. et al. Analyzing and improving the image quality of stylegan. *CoRR*, abs/1912.04958, 2019.

[45] Wu, J., Zhang, C., Xue, T. et al. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *CoRR*, abs/1610.07584, 2016.

[46] Lim, J.J., Pirsiavash, H. and Torralba, A. Parsing ikea objects: Fine pose estimation. In *2013 IEEE International Conference on Computer Vision*, pages 2992–2999, 2013. doi:10.1109/ICCV.2013.372.

[47] Kwon, G., Han, C. and shik Kim, D. Generation of 3d brain mri using auto-encoding generative adversarial networks, 2019.

[48] Yang, J., Liu, S., Grbic, S. et al. Class-aware adversarial lung nodule synthesis in ct images. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 1348–1352, 2019. doi:10.1109/ISBI.2019.8759493.

[49] Xie, S., Girshick, R.B., Dollár, P. et al. Aggregated residual transformations for deep neural networks. *CoRR*, abs/1611.05431, 2016.

[50] Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S. et al. 3d u-net: Learning dense volumetric segmentation from sparse annotation. *CoRR*, abs/1606.06650, 2016.

[51] Salimans, T., Goodfellow, I.J., Zaremba, W. et al. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[52] Szegedy, C., Vanhoucke, V., Ioffe, S. et al. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[53] Che, T., Li, Y., Jacob, A.P. et al. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016.

[54] Heusel, M., Ramsauer, H., Unterthiner, T. et al. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.

[55] Wang, Z., Bovik, A.C., Sheikh, H.R. and Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. ISSN: 10577149. doi:10.1109/TIP.2003.819861.

[56] Wang, Z., Simoncelli, E.P. and Bovik, A.C. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, volume 2, pages 1398–1402 Vol.2, 2003. doi:10.1109/ACSSC.2003.1292216.

[57] Geman, D., Geman, S., Hallonquist, N. and Younes, L. Visual turing test for computer vision systems. *Proceedings of the National Academy of Sciences*, 112(12):3618–3623, 2015. ISSN: 0027-8424. doi:10.1073/pnas.1422953112.