Technical University of Košice
Faculty of Electrical Engineering and Informatics

Generative adversarial network for augmenting insufficient medical datasets

Master's Thesis

2024 Bc. Miroslav Plavec

Technical University of Košice
Faculty of Electrical Engineering and Informatics

Generative adversarial network for augmenting insufficient medical datasets

Master's Thesis

Study Programme:	Intelligent Systems
Field of study:	Computer Science
Department:	Department of Cybernetics and Artificial Intelligence (KKUI)
Supervisor:	doc. Ing. Marek Bundzel, PhD.
Consultant(s):	Ing. Maroš Hliboký

Košice 2024 **Bc. Miroslav Plavec**

Abstract

Lung ultrasound, a rapidly growing imaging technique, offers real-time visualization of the lungs without radiation exposure. Accurate lung disease diagnosis relies on identifying specific artifacts like A-lines and B-lines that can be seen on ultrasound images. This thesis addresses the issue of limited training data for deep learning models that assist in interpreting these artifacts. We propose a solution using Generative Adversarial Networks (GANs) to generate synthetic lung ultrasound images containing A-lines and B-lines. In our experiments, three different types of GANs (DCGAN, WGAN-GP, and Pix2Pix) were implemented to generate these synthetic images. Each with variations in network architecture and loss function. Augmenting the training dataset with generated images resulted in a 2% improvement in A-line classification and a significant 5% improvement in B-line classification. This thesis demonstrates the potential of GANs to address data scarcity limitations in medical imaging, paving the way for a broader improvement in the accuracy of diagnosis and treatment for various diseases, ultimately benefiting patient care.

Keywords

medical image processing, lung ultrasound, deep learning, generative adversarial networks, synthetic data augmentation

Abstrakt

Ultrazvuk pľúc je rýchlo rozvíjajúca sa technika, ktorá dokáže zachytiť snímok pľúc v reálnom čase, bez vystavenia škodlivému žiareniu. Presná diagnostika pľúcnych chorôb sa spolieha na identifikáciu špecifických artefaktov, ktoré sa nachádzajú na týchto ultrazvukových snímkach. Príkladom takýchto artefaktov sú A-línie alebo B-línie. Naša práca sa zaoberá problémom obmedzených trénovacích dát pre modely hlbokého učenia, ktoré pomáhajú pri interpretácii týchto artefaktov. V tejto práci

navrhujeme riešenia, ktoré sú schopné generovať nové ultrazvukové snímky pľúc obsahujúce A-línie a B-línie. Dané riešenia sú založené na generatívne súperiacich sieťach (GANs). V našich experimentoch sme implementovali tri rôzne typy GANs (DCGAN, WGAN-GP a Pix2Pix) na vytvorenie syntetických snímok, pričom každý z nich sa odlišuje architektúrou siete a použitou chybovou funkciou. Obohatenie pôvodnej trénovacej množiny našimi vygenerovanými snímkami viedlo k 2% zlepšeniu v klasifikácii A-línii, a 5% zlepšeniu v klasifikácii B-línii. Výsledky tejto práce poukazujú na potenciál GANs pri riešení problému obmedzeného množstva trénovacích dát v medicínskej oblasti.

Kľúčové slová

spracovanie medicínskych obrazov, ultrazvuk pľúc, hlboké učenie, generatívne súperiace siete, augmentácia syntetickými dátami

71657

TECHNICAL UNIVERSITY OF KOŠICE

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATICS

Department of Cybernetics and Artificial Intelligence

DIPLOMA THESIS ASSIGNMENT

Field of study: **Computer Science**

Study programme: **Intelligent Systems**

Thesis title:

Generative adversarial network for augmenting insufficient medical datasets

Generatívna sieť na rozšírenie nedostatočných lekárskych súborov údajov

Student: **Bc. Miroslav Plavec**

Supervisor: **doc. Ing. Marek Bundzel, PhD.**

Supervising department: **Department of Cybernetics and Artificial Intelligence**

Consultant: **Ing. Maroš Hliboký**

Consultant`s affiliation: **Department of Cybernetics and Artificial Intelligence**

Thesis preparation instructions:

1. Create a literature review on the issues of Generative Adversarial Networks focusing on medical data.
2. Write an overview of existing solutions of Generative Adversarial Networks, focusing on generating medical data and their subsequent utilization.
3. Propose optimal data preprocessing techniques.
4. Utilize processed insights in the implementation of your solution.
5. Compare the results of individual models and conduct appropriate evaluations.
6. Prepare documentation according to the supervisor's instructions.

Language of the thesis: English

Thesis submission deadline: 19.04.2024

Assigned on: 31.10.2023

TECHNICKÁ UNIVERZITA V KOŠICIACH FAKULTA ELEKTROTECHNIKY A INFORMATIKY -11-

prof. Ing. Liberios Vokorokos, PhD.

Dean of the Faculty

Declaration

I hereby declare that this thesis is my own work and effort. Where other sources of information have been used, they have been acknowledged.

Košice, April 19, 2024 .....

Signature

Acknowledgement

I would like to express my sincere thanks to my supervisor doc. Ing. Marek Bundzel, PhD, the main Supervisor. Special mention should go to Ing. Maroš Hliboký, for his constant, and constructive guidance throughout the study. To all other who gave a hand, I say thank you very much.

Contents

List of Figures

List of Tables

List of Terms

LUS - Lung ultrasound

GANs - Generative Adversarial Networks

FID - Frechet Inception Distance

DCGAN - Deep Convolutional Generative Adversarial Network

ProGAN - Progressive Growing GAN

LAPGAN - Laplacian GAN

MRI - Magnetic Resonance Imaging

CNN - Convolutional Neural Network

WGAN - Wasserstein GAN

WGAN-GP - Wasserstein GAN with Gradient Penalty

WGAN-GP-NM - Unconditional WGAN-GP, no mask is used as input

WGAN-GP-ED - Conditional WGAN-GP, Encoder-Decoder type generator

WGAN-GP-SC - Conditional WGAN-GP, Encoder-Decoder type generator with skip connection

Introduction

The history of lung ultrasound (LUS) traces back to the early 1960s, with the initial use of ultrasound waves for medical imaging. Since then, advancements in scientific discoveries and computing power have fueled the rapid development of LUS technology. LUS has become an important tool in diagnosing and managing respiratory illnesses due to its portability, affordability, and real-time imaging capabilities. By utilizing high-frequency sound waves, lung ultrasound captures images revealing crucial anatomical features within the lungs, including the pleura and specific artifacts. Among these artifacts, A-lines and B-lines hold particular significance, as they provide valuable information about the patient's lung condition.

Identifying and interpreting these ultrasound artifacts can be challenging and requires expertise. To assist doctors, deep learning algorithms could be developed to automate the detection and analysis of lung ultrasound patterns. However, a major challenge in leveraging lung ultrasound for advanced analysis using deep learning models lies in the limited availability of training data. Traditional augmentation techniques offer partial solutions but fail to generate entirely new samples, thereby restricting the diversity and size of the dataset.

This thesis explores the potential of Generative Adversarial Networks (GANs) to address this data scarcity issue. GANs are a class of deep learning models adept at creating new data samples that closely resemble the original dataset. By employing different types of GANs, we aim to generate high-quality synthetic lung ultrasound images of realistic-looking A-lines and B-lines artifacts. We employed DCGAN, WGAN-GP, and Pix2Pix to compare their abilities to generate high-quality LUS images. To determine which type can generate the highest quality images, we compare the classifier's performance when trained solely with real images to its performance when trained on a dataset consisting of both real images and generated images.

Our thesis is organized into several sections. In the second section, we concentrate on

lung ultrasound. Here, we delve into the history of this technology and its advantages over traditional lung imaging methods. We describe how sound waves are employed to capture images of the internal lung structure and discuss the significance of A-lines and B-lines, and their interpretation by medical professionals. In the third section, we provide an overview of GANs. We describe the underlying concept behind GANs and explain how this type of generative models can learn the distribution of real data and utilize it to generate synthetic images. In the fourth section, we introduce several studies that have focused on image synthesis in the medical domain. In the fifth section, we offer more detailed information about our dataset and discuss all the types of GANs utilized in our experiments. For each type, we also provide a detailed description of the network architectures employed. In our final section, we present all our experiments and the corresponding results. We then proceed to compare the performance of different GAN types based on their effectiveness in enhancing the classification model's performance using the generated images.

1 The problem expression

1. Create a literature review on the issues of Generative Adversarial Networks focusing on medical data.
2. Write an overview of existing solutions of Generative Adversarial Networks, focusing on generating medical data and their subsequent utilization.
3. Propose optimal data preprocessing techniques.
4. Utilize processed insights in the implementation of your solution.
5. Compare the results of individual models and conduct appropriate evaluations.
6. Prepare documentation according to the supervisor's instructions.

2 Lung ultrasound

This section provides an overview of lung ultrasound fundamentals. It covers the principles of image acquisition using ultrasound waves, outlines the characteristics of ultrasound probes suitable for this purpose, and discusses common lung ultrasound artifacts, especially A-lines and B-lines.

Ultrasound technology has become increasingly important in medical imaging since its introduction in the early 1960s. Its development has been rapid, thanks to scientific discoveries and advances in computing. When ultrasound was first used for regional anesthesia in the 1990s, the output was a chart of dots. However, it now provides a real-time image that can be easily understood in relation to the anatomy [1]. Lung ultrasound (LUS) is a low-cost, radiation-free, and easy-to-use technique that uses sound waves to create an image of the lungs. It has transformed the way healthcare professionals diagnose and manage respiratory conditions. Unlike traditional chest X-rays, lung ultrasound offers a quick and safe way to look directly into the lungs at the bedside. This makes it a valuable tool in emergency departments, intensive care units, and even for use by paramedics in the field [2].

2.1 Lung ultrasound probes and image formation

Ultrasound waves are sound waves that have frequencies above the upper limit of human hearing, which is around 20 KHz. Medical professionals use diagnostic ultrasound with frequency in a range between 2 and 20MHz. The ultrasound machine generates these waves using piezoelectric crystals that are located in the head of the probe. When electrical energy is passed through the piezoelectric crystals of an ultrasound machine's probe, the crystals deform and generate ultrasound waves [3]. The appearance of lung ultrasound images is determined by the amount of air and fluid present in the lung tissue. This is due to the phenomenon of acoustic impedance, which measures the resistance of particles in a medium to mechanical vibrations. The resistance increases in proportion to the density of the medium and

the propagation velocity of ultrasound in the medium [4].

During a lung examination, a handheld probe emits high-frequency sound waves that travel through the chest wall. These waves interact with different tissues in their path, bouncing back when they encounter an interface between tissues with varying densities. Air in the lungs has a low density compared to other tissues, allowing most of the sound waves to pass through. However, the interface between air-filled lungs and the fluid-filled pleura (lining of the lungs) reflects the sound waves back to the probe. The returning echoes are then processed by the ultrasound machine and converted into an image displayed on a screen [3].

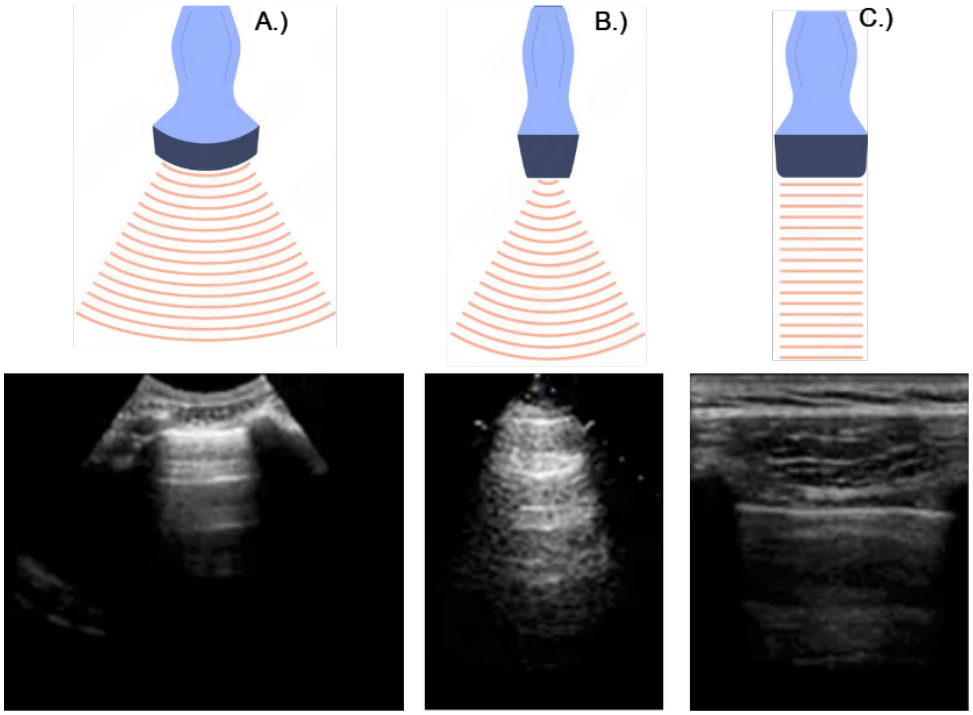

Figure 2 – 1 A.) Curvilinear probe, B.) Phased array probe, C.) Linear probe

There are three main probes, that could be used to capture ultrasound images of lungs. These probes come in various sizes and shapes, each with its own set of pros and cons. They use the same principle and differ in the footprint, scanning frequency, and scanning depth. Curvilinear probe (3–5 MHz) releases a curved arrangement of ultrasound beams that forms a curved footprint. Although it has a lower frequency,

it covers a wider area of imaging and provides greater depth. However, the image quality is not as good as that produced by a linear probe, as the resolution is grainy. Despite its grainy resolution, it is the preferred transducer for imaging and interventions that require deeper blocks [3]. Phased array probe (3–4.5 MHz) releases ultrasound beams, which are arranged in a "stacked" construction, also known as a phased array. It has a similar characteristic footprint as the curvilinear probe, but it is smaller, making it useful for getting in between the ribs [4]. Linear probe (8–12 MHz) is named as such because it has a linear piezoelectric arrangement. It emits a linear array of ultrasound waves and produces a rectangular footprint. It is usually of high frequency and low penetration. The higher frequency gives the linear probe better image resolution, which makes it useful for many applications where high-quality images are needed [3]. For all our experiments, we will use images that were captured by a linear probe.

2.2 Lung ultrasound artifacts

LUS artifacts are visual disturbances in the ultrasound image that arise from the reflection, refraction, or scattering of sound waves by structures within the lungs or at the lung surface. These artifacts themselves are not a direct representation of lung anatomy, but rather how sound waves interact with the lung tissue. However, interpreting these artifacts is an important part of lung ultrasound because they can provide valuable clues about the underlying lung pathology [5]. There are several different types of lung ultrasound artifacts, but in our thesis, we will focus on A-lines and B-lines. The presence or absence of A-lines and B-lines can provide doctors with information about a patient's lung condition and the presence of any diseases. A crucial role in the formation of both A-lines and B-lines plays the pleura. The pleura refers to the thin, double-layered membrane that surrounds the lungs and lines the inner chest wall. It consists of two layers: the visceral pleura, which directly covers the lung surface, and the parietal pleura, which lines the inner surface of the chest

wall. During ultrasound examination, the pleura appears as a bright line known as the pleural line. This line serves as a crucial anatomical landmark in lung ultrasound, providing a reference point for evaluating lung pathology [6] [7].

2.2.1 A-lines

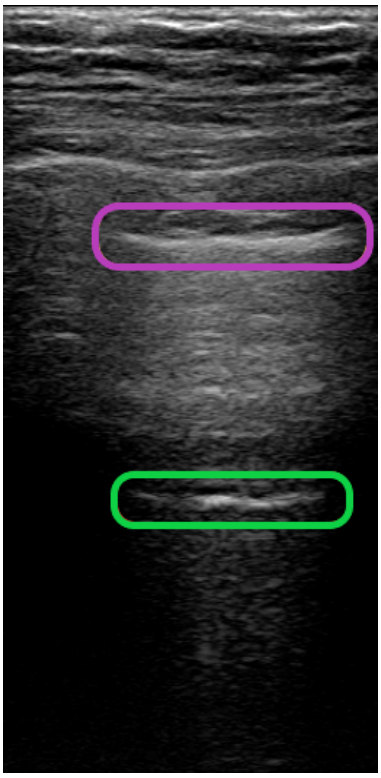

Figure 2 – 2 Example of A-line, purple box - pleura, green box - A-line

A-lines are echogenic, gradually fading horizontal lines arranged at equal intervals below the pleural line. They represent the repetition artifact of the parietal pleura. When there is air below the pleura, ultrasound waves reflect back to the probe, causing the movement of waves back-and-forth between the probe and the air beneath the pleura. This causes the appearance of A-lines. The presence of A-lines is not a sign of pathological finding and is generally associated with healthy, air-filled lungs. Conversely, the absence of A-lines doesn't necessarily indicate a problem, as they may not always be present on the ultrasound image [8].

2.2.2 B-lines

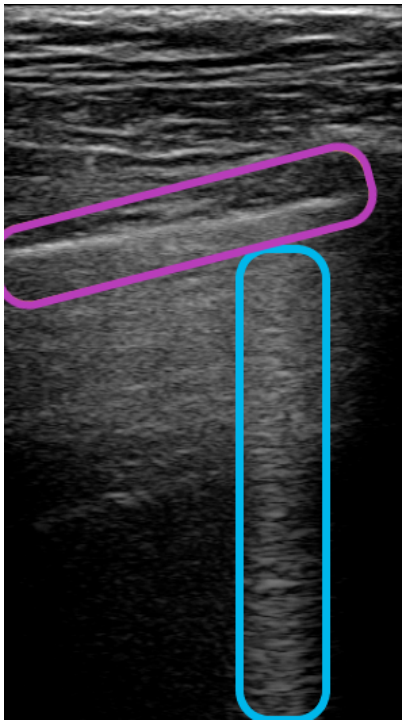

Figure 2 – 3 Example of B-line, purple box - pleura, blue box - B-line

B-lines, also known as ultrasound lung comets, are vertical reverberation artifacts that originate from the pleural line and extend to the depth of the image without decreasing in intensity. They move synchronously with lung sliding and erase or cancel out the A-lines. An increase in lung tissue density, caused by substances like exudate, transudate, collagen, or blood, reduces the acoustic mismatch between pleura and underlying lung structures. This causes the reflection of the ultrasound beam back to the probe, producing distinctive comet-tail artifacts. B-lines are usually absent under normal conditions and present in alveolar-interstitial syndromes such as pulmonary edema or pneumonia [8].

3 Generative adversarial networks

Generative models play a key role in the field of artificial intelligence. Their primary purpose is to understand and capture the underlying patterns or distributions from a given set of data. Generative models come in various forms, each with its unique approach to understanding and generating data. Some of the most prominent examples of generative models are Variational Autoencoders, Pixel Recurrent Neural Networks, Diffusion models, or Generative Adversarial Networks [9]. In this section, we'll describe how Generative Adversarial Networks can create realistic images within a particular domain. We'll break down their key components and the objective function that fuels their training process. Finally, we'll describe some of the biggest challenges in training GANs.

3.1 An introduction to GANs

Generative Adversarial Networks, or GANs for short, have revolutionized the way we approach tasks like image generation, data augmentation, and domain adaptation. Ian Goodfellow and his colleagues introduced GANs in 2014, and since then, they have gained enormous attention. GANs can generate highly realistic and diverse data samples that mimic the distribution of real-world data. The basic idea behind GANs is very simple. The architecture consists of two neural networks, a generator, and a discriminator as can be seen in Figure 3 – 2. These two networks engage in an adversarial game, where the generator seeks to produce increasingly realistic samples to deceive the discriminator, while the discriminator attempts to become more adept at distinguishing real from fake samples [10].

From their humble beginnings in 2014, GANs have evolved into a diverse family of models with applications spanning image generation, text synthesis, audio generation, and beyond. While the core concept remains the same, GANs have witnessed significant progress in the last decade. Researchers have experimented with different loss functions and network architectures to overcome limitations and achieve bet-

ter outcomes. In Figure 3 – 1, you can see various types of GANs that have been introduced over the years [11].

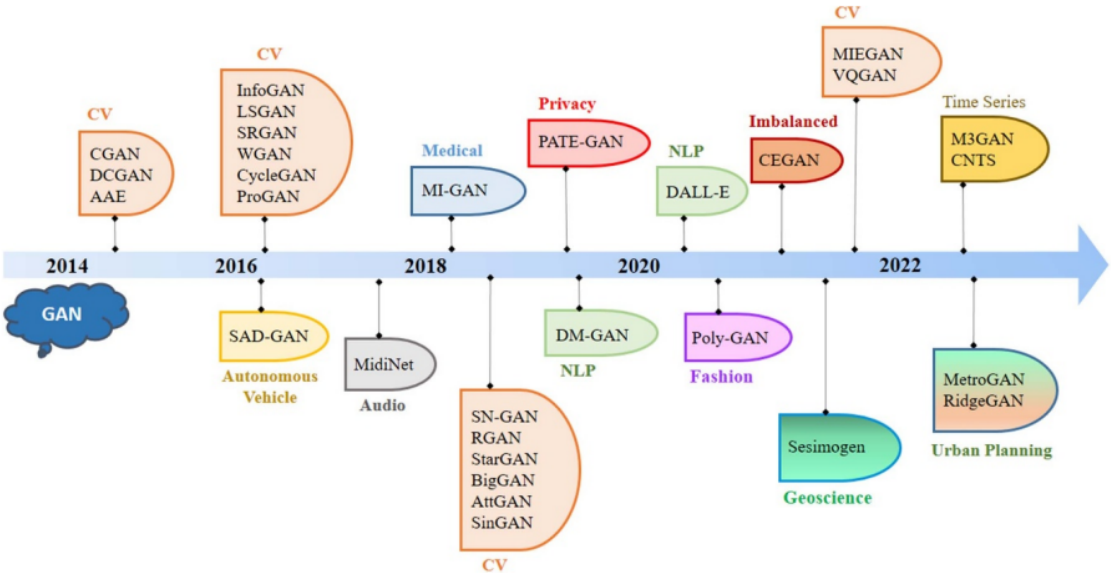

Figure 3 – 1 The Evolving Landscape of GANs [11]

3.2 Generator and discriminator

The generator operates as a differentiable function, typically implemented as a neural network. It takes a random noise vector of fixed length as input and transforms it into a sample resembling data from a specific domain. This noise vector is drawn from a predefined probability distribution, for example, gaussian distribution, and serves as a source of noise for the generative process. Without random input, the generator would produce the same synthetic data repeatedly. The input is a vector of random numbers. It is not an image or a flattened image and has no meaning other than the meaning applied by the generator model. However, after training, specific points within this multidimensional noise space will map to corresponding points in the target data domain, effectively forming a compressed representation of the underlying data distribution [12].

The discriminator, also a neural network, takes an example (either real or generated)

from the problem domain as input and attempts to classify it as real or generated. Real examples come from the training dataset, while generated examples are produced by the generator model. During training, the discriminator acts like a regular classification model. After the training process, the discriminator model can be discarded as we are interested in the generator [12].

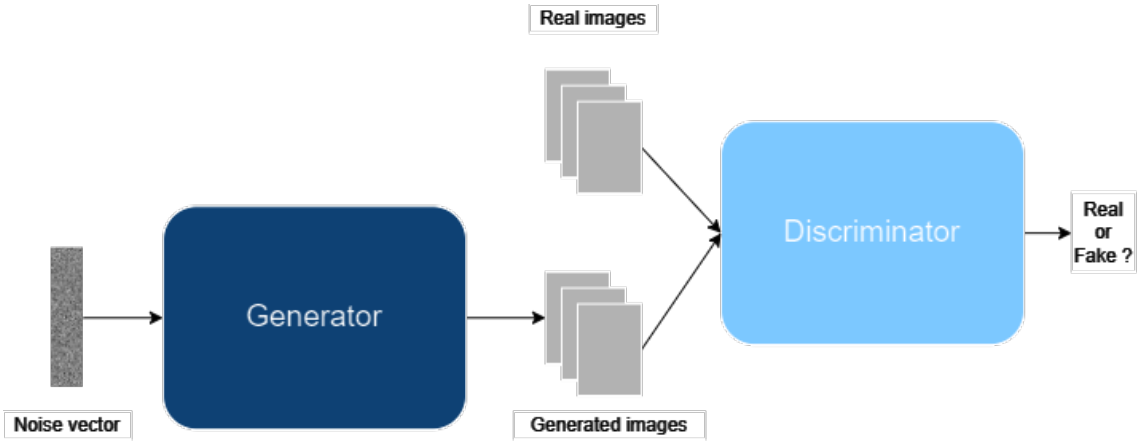

Figure 3 – 2 The architecture of vanilla GAN

3.3 Competitive training in GANs

Generative modeling is an unsupervised learning problem, although a clever property of the GANs architecture is that the training of the generative model is framed as a supervised learning problem. The two models, the generator and discriminator, are trained together. The generator creates a batch of samples, and these, along with real examples from the domain, are provided to the discriminator and classified as real or fake. The discriminator is then updated to improve its ability to distinguish real and fake samples in the next round. Importantly, the generator is updated based on how well it fools the discriminator [10]. They are adversarial in the game theory sense, playing a zero-sum (min-max) game. In a zero-sum game, when the discriminator successfully identifies real and fake samples, it's rewarded (no change needed to its parameters). Conversely, the generator is penalized with updates to its parameters. On the other hand, when the generator fools the discriminator, it's

rewarded (no change needed to its parameters), while the discriminator is penalized, and its parameters are updated. The objective function used to train a GANs is pressented in Equation 3.1. The generator seeks to minimize this function and the discriminator aims to maximize it. In other words, the discriminator wants to maximize the difference between its classification scores for real and fake data, and generator wants to minimize this score [12].

$$\min_G \max_D V(D,G) = E_{x\sim p_{\text{data}}(x)}[\log D(x)] + E_{z\sim p_z(z)}[1 - \log D(G(z))] \tag{3.1}$$

where:

G = generator

D = discriminator

x = training dataset

z = random variable

p_{data} = real data distribution

p_z = noise distribution

The first term $E_{x\sim p_{\text{data}}(x)}[\log D(x)]$ in Equation 3.1 represents the expected value of the logarithm of the discriminator's output when evaluated on real data. And the second term $E_{z\sim p_z(z)}[1-\log D(G(z))]$ represents the expected value of the logarithm of (1 - discriminator's output) when evaluated on generated data. The discriminator uses both terms to updates its parameters, while the generator uses only second term to updates its parameter [13].

There are two main approaches for data generation with GANs: unconditional and conditional. Unconditional synthesis involves generating samples from random noise without any additional information. The generator network learns to capture the underlying data distribution of the training dataset without any guidance such as

class labels. In this approach, the standard GANs objective function is used, which can be seen in Equation 3.1. On the other hand, in conditional synthesis, both the generator and discriminator are conditioned on additional information such as class labels, attribute vectors, or even other data samples. The generator network takes both random noise vector and conditioning information as input and produces samples that conform to the specified condition. Conditional GANs are widely used in tasks where specific attributes or conditions need to be controlled during generation, such as image-to-image translation, text-to-image synthesis, and image inpainting. In this approach, a slightly modified GANs objective function is used, which can be seen below.

$$\min_G \max_D V(D,G) = E_{x\sim p_{\text{data}}(x)}[\log D(x|y)] + E_{z\sim p_z(z)}[1 - \log D(G(z|y))] \quad (3.2)$$

where y represents conditional information [14].

3.4 Challenges in training GANs

Despite the remarkable capabilities of GANs, training them remains a notoriously challenging task. There are several common problems specifically associated with GANs training.

3.4.1 Mode collapse

One of the most common challenge in GANs training is mode collapse, where the generator fails to capture the full diversity of the data distribution and instead gets stuck producing a limited set of outputs. This can happen due to factors such as a weak discriminator, unbalanced learning rates, high-dimensional data, sparse data distribution, or architectural limitations. As a result, the diversity of generated samples diminishes, impacting the quality and creativity of the generated data [15]. To address mode collapse, researchers have proposed several strategies.

One approach is to modify the loss functions used during training to encourage the generator to explore a wider range of output modes. For example, the Wasserstein GAN objective function penalizes the generator for generating samples that are dissimilar to real data, thereby incentivizing diverse sample generation [16].

3.4.2 Training instability

Another problem is that GANs training can be unstable, characterized by oscillations and divergence in the learning process. This instability can manifest as vanishing gradients, where the generator or discriminator fails to learn effectively, leading to slow convergence or failure to converge altogether [15]. To address this issue, researchers have proposed techniques such as gradient clipping and feature matching to stabilize GANs training and improve convergence. Gradient clipping limits the magnitude of gradients during backpropagation, preventing them from exploding or vanishing and stabilizing the training process [16]. Feature matching involves matching the feature representations of real and generated samples, encouraging the generator to produce samples that are indistinguishable from real data [17].

3.4.3 Evaluation

Unlike other deep learning models trained using a single loss function until convergence, a GANs takes a unique approach. The generator is trained indirectly through a competitive process with a discriminator that learns to classify images as real or generated. Consequently, there's no single objective loss function solely for the generator, and evaluating progress based solely on discriminator‘s loss can be misleading [10].

Probably the best approach to evaluate generated images would be by a manual inspection, but this has some limitations. It requires knowledge of what is realistic and what is not for the target domain. It is also limited by the number of images that can be reviewed in a reasonable time, making it difficult to use during model

development. Due to these limitations, a suite of qualitative and quantitative techniques have been developed to assess the performance of a GANs model based on the quality and diversity of the generated images. We have chosen to use Fréchet Inception Distance (FID) for evaluating our experiments. Although there are other evaluation metrics available, there is no clear hierarchy among them and some metrics could even provide conflicting results. To avoid any confusion and contradictory interpretations, we have decided to use only one metric [18].

The FID starts by extracting features from a pre-trained deep convolutional neural network. Typically, the Inception model, which is trained on a large dataset such as ImageNet, is used for this purpose. Both real and generated images are fed into a pre-trained model, which acts as a feature extractor. Next, for both sets of images, the feature statistics, mean and covariance matrix are computed using the extracted features from the Inception model. Then, the Fréchet distance, which is a measure of dissimilarity between two probability distributions, is calculated by comparing the feature statistics of the real and generated images. A lower FID score indicates that the generated images have high-level features similar to the real images, implying better quality and diversity in the generated images [19].

Mathematically the FID is expressed as:

$$FID = |\mu_r - \mu_g|^2 + Tr\left(C_r + C_g - 2\sqrt{C_r C_g}\right) \tag{3.3}$$

where (μ_r, C_r) and (μ_g, C_g) represent the mean and covariance matrix pair for the real images and the generated images, respectively. Tr represent the trace of matrix, which is the sum of matrix diagonal entries [19].

4 Related work

Medical image synthesis is the most successful application of GANs in medical image analysis so far. It can help to address the challenges of insufficient medical images available or imbalanced data categories. Traditional data augmentation techniques such as image cutting, flipping, and rotation can only modify the data in terms of direction or size, but don't generate new data. In contrast, GANs technology can create completely new data [20]. In this section, we will introduce several studies on GANs used for generating medical images. This section is divided into two subsections. In the first subsection, we will describe four studies focused on unconditional synthesis. In the second subsection we will describe four studies focused on conditional synthesis. Each subsection will begin with a table summarizing these studies.

4.1 Unconditional approach

Paper	Domain	Scans number	Model	Evaluation
[21]	Obstetric US	8,747	StyleGan2	precision, recall
[22]	Breast US	1,057	DCGAN	expert assessment
[23]	Skin lesions	10,000	ProGAN	expert assessment
[24]	Brain MRI	49,452	WGAN-GP	accuracy

Tab. 4 – 1 Summary of unconditional studies

4.1.1 GANs to improve fetal brain classification

Alberto Montero and his co-workers conducted a study to explore the potential of GANs in improving the classification of fine-grained planes within the fetal brain. They investigated whether the accuracy of classifying specific anatomical planes within the fetal brain could be enhanced by incorporating synthetic ultrasound images generated by GANs. To conduct their experiment, they used a dataset of real

ultrasound images depicting various fetal brain planes. The final dataset used for the study included 8,747 images, which were cropped before being used to train the generative model. The researchers employed StyleGAN2-ADA, a powerful GAN architecture capable of creating realistic images, to generate synthetic images for the experiment. The effectiveness of the GAN-generated data was evaluated by comparing the performance of two classification models. One of the models was trained solely on real ultrasound images, while the other was trained on a combined dataset that included both real and synthetic images generated by StyleGAN2-ADA. To assess the performance of each classifier in differentiating between various fetal brain planes, metrics like accuracy and area under the curve were used. The results indicated that augmentation via StyleGAN2-ADA resulted in a 1.6% improvement over the accuracy baseline and a 1.7% improvement over the area under the curve baseline [21].

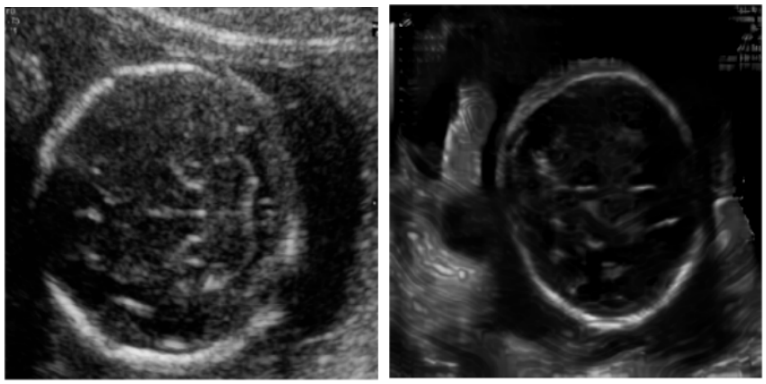

Figure 4 – 1 Real image(left), Generated image(right)

4.1.2 Breast ultrasound image synthesis using DCGAN

Authors of this study investigated the use of a DCGAN to synthesize realistic breast ultrasound images. Two experienced breast radiologists then evaluated these images from a clinical perspective. The training data consisted of 528 images of benign masses and 529 images of malignant masses from an ultrasound DICOM dataset.

The DCGAN employed a strided convolutional architecture with batch normalization and LeakyReLU activations in the discriminator, while the generator utilized transpose-convolutional layers with batch normalization and ReLU activations. After training the model, the authors generated 20 images each from 50, 100, 200, 500, and 1000 epoch. These generated images, along with 40 original images, were then subjectively evaluated by the two radiologists. The model achieved the best results after being trained for 500 epochs. At this point, 37.5% of the images were either misinterpreted as being original or were indistinguishable from the original images. Interestingly, 14.0% of the original images were also indistinguishable from the synthesized images [22].

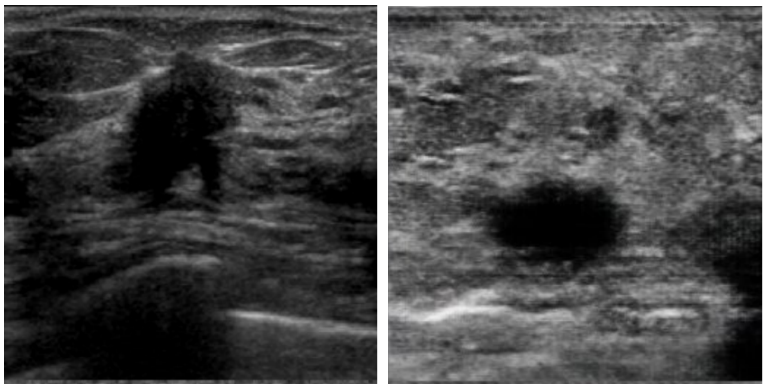

Figure 4 – 2 Real image(left), Generated image(right)

4.1.3 Generating images of skin lesions with GANs

Christoph Baur and his colleagues aimed to create synthetic images that realistically represent a diverse range of skin lesions. They used a concept called Progressive Growing GAN (ProGAN) to generate these images. In ProGAN, the generator and discriminator are trained progressively in a step-by-step manner, starting from low-resolution images and gradually increasing the resolution. This progressive training approach allows for more stable training and enables the generation of high-quality, high-resolution images. The researchers used the ISIC2018 dataset containing 10,000 dermoscopic images of both benign and malignant lesions. To assess the realism of

the generated images, the researchers conducted a visual turing test involving 3 dermatologists and 5 deep learning experts. Each participant was asked to classify the same random mix of generated and real images as either real or fake. Using the ProGAN, they first generated 30 synthetic images, which were then mixed with 50 randomly chosen images from the real training dataset. The results showed that, on average, 56% of the fake images were mistaken as real, and 42% of the real images were mistaken as fake [23].

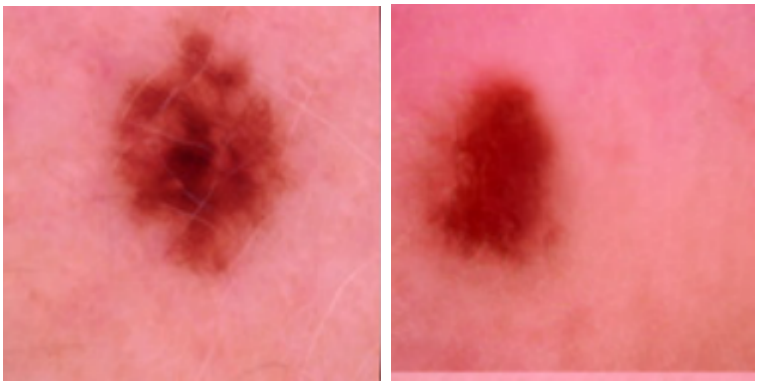

Figure 4 – 3 Real image(left), Generated image(right)

4.1.4 GANs as an augmentation technique for MRI data

The study conducted by Konidaris and his coworkers aimed to investigate the effectiveness of Wasserstein GAN with Gradient Penalty for data augmentation in MRI analysis. They conducted experiments using the ADNI dataset consisting of scans of normal patients and patients with Alzheimer's disease. They had 25,154 normal and 24,298 Alzheimer's disease scans available for training. They trained two GANs, one for each class. The generator architecture had 11 layers with more than 15 million trainable parameters. They used three types of layers: fully connected, convolutional, and transpose convolutional. The input to the generators was a vector of 128 random values in the range [0,1). The discriminator was a regular CNN architecture aimed towards binary classification, consisting of 11 layers and around 9.5 million trainable parameters. Once the GANs were fully trained, they

were used to produce 50,000 fake images for each class. These images were used to help train the classifier. The results showed that by adding generated images into original training dataset, there was an increase of 11.68% in accuracy [24].

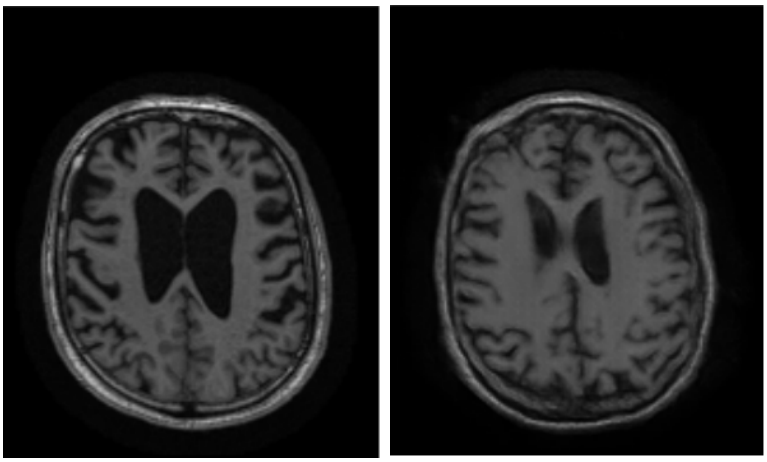

Figure 4 – 4 Real image(left), Generated image(right)

4.2 Conditional approach

Paper	Domain	Scans number	Model	Evaluation
[25]	Chest US	3,261	PGSgan	FID
[26]	Vessel images	150	DCGAN+VGG19	FID, SSIM
[27]	Skin lesion	2,150	Pix2Pix	dice score
[28]	Mammography	443,556	ProGan+WGAN-GP	expert assessment

Tab. 4 – 2 Summary of conditional studies

4.2.1 Synthesis of ultrasound umages via ProGAN

The authors of this paper have developed a new framework that can produce high-resolution, realistic, and editable ultrasound images in an efficient manner. To

improve the structure fidelity and simplify the synthesis process, they used a Sketch GAN (Sgan) to add supplementary sketch guidance to the object mask in a conditional GAN. They designed the Sgan generator as an encoder-decoder architecture with a 10-residual blocks, and the discriminator followed a PatchGAN design with a 30x30 output. They used the canny algorithm to extract binary edges effectively as a sketch and concatenated the original mask with the edge sketch without affecting the target object's area. They started the training of the progressive-growing Sgan on low-resolution conditional input. Then they progressively increased the input resolution and added learnable layers to the existing network with shared weights for continuous training. All of their experiments were carried out on a dataset of 3,261 ovarian US images. They used 2,848 images for training and the remaining 431 images for testing. They evaluated their experiments using FID and showed that their approach is better than a simple conditional DCGAN [25].

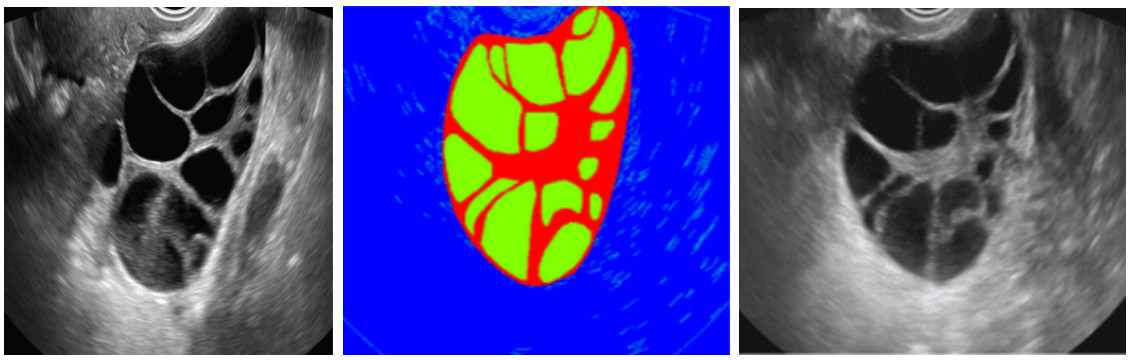

Figure 4 – 5 Real image(left), Input mask(middle), Generated image(right)

4.2.2 Synthesizing retinal and neuronal images with GANs

The aim of this paper was to create multiple realistic-looking retinal images from an unseen tubular structured annotation that contains the binary vessel morphology. The authors of this study have proposed a framework called Tub-sGAN, which is the first study to incorporate style transfer into a GANs framework. For the generator, they have used an encoder-decoder strategy with skip connections. The

generator takes a 400-dimensional noise and a binary segmentation map as inputs. The generator was built with multiple components, which composed of convolutional layer, batch normalization layer, and LeakyReLU activation, except the last one, which was Tanh. The same components were used in the discriminator. To make the generated images diverse, they have used the VGG-19 neural network to extract style (texture) from the target image to the generated image. To make it work, they have added style loss and content loss to the vanilla GANs loss. All of their experiments have been carried out on a dataset of 150 images. They have evaluated their experiments with metrics like SSIM and FID and their experiments have shown that Tub-sGAN can generate realistic-looking images [26].

Figure 4 – 6 Real image(left), Input mask(middle), Generated image(right)

4.2.3 Mask2Lesion: skin lesion image synthesis

Kumar Abhishek and Ghassan Hamarneh used a lesion masks to generate synthetic lesion images that can be used to augment the segmentation training dataset. They modeled image synthesis as an image-to-image translation task and developed a deep neural network model called Mask2Lesion to generate the synthetic data. To generate the synthetic data, they used lesion segmentation masks as input to their generative algorithm. They chose to use U-net with L1 loss as the generator and PatchGAN as the discriminator, which is a convolutional neural network that divides the image into several patches and labels them as real or fake. The dataset used

for training contained 2,000 training images and 150 test images. They trained the model for about 200 epochs. Their results showed a significant improvement in the segmentation accuracy when the training dataset for the segmentation network was augmented with generated images. Segmentation network achieved an improvement of 5.17% in the mean dice score compared to a model trained with only classical data augmentation techniques [27].

Figure 4 – 7 Real image(left), Input mask(middle), Generated image(right)

4.2.4 Mask embedding for medical image synthesis

A study conducted by Yinhao Ren and his team aimed to investigate the effectiveness of GANs for data augmentation in mammography analysis. The team used a mammography dataset consisting of 443,556 images. They proposed a novel approach to improve the generation of realistic high-resolution medical images with semantic control. They used a U-Net style generator that takes both a latent vector (gaussian noise vector) and a semantic mask. To transform a binary mask to a 32-dimensional vector, they used seven convolutional layers. They then concatenated this vector with a 100-dimensional vector of random numbers. Finally, they upsampled this final 132-dimensional vector with seven deconvolution layers. They used a progressive training strategy and Wasserstein GAN with Gradient Penalty loss. After training, they randomly picked 50 real breast masks and generated mammograms. Two expert radiologists were then asked to rate each mammogram. The

results confirmed that their approach with mask embedding provided a considerable improvement in realism [28].

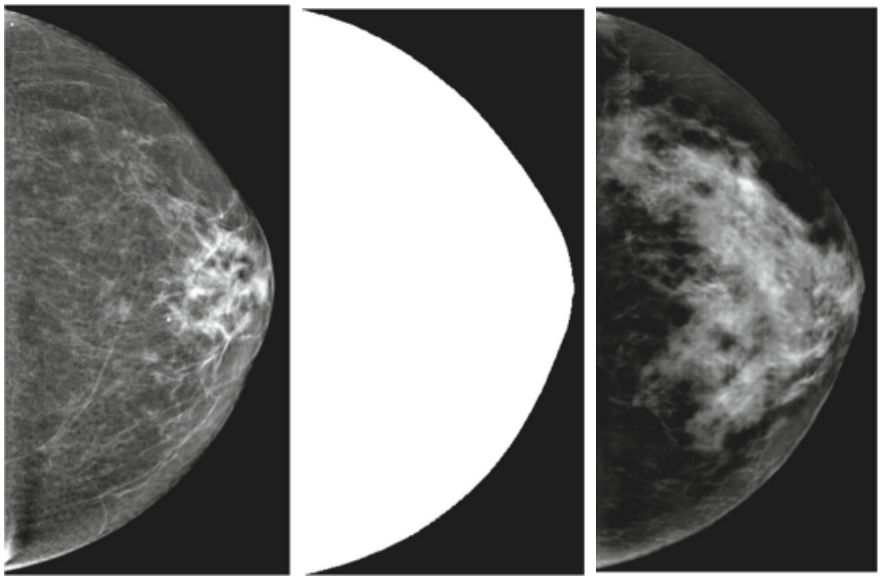

Figure 4 – 8 Real image(left), Input mask(middle), Generated image(right)

5 Data and methods

In this section of our thesis, we will provide detailed information about the data and methods we used in our experiments. Firstly, we will describe our data, followed by a description of the GANs we employed. In our experiments, we utilized three types of GANs: DCGAN, WGAN-GP, and Pix2Pix.

5.1 Dataset

The LUS images used in our experiments were obtained from the Department of Thoracic Surgery and the Clinic of Radiology at Jessenius Faculty of Medicine in Martin. The data collection process was carried out in close collaboration with medical experts from this institution to ensure that only images captured by proficient specialists were included. All of the images comes from videos captured during a lung examination. These videos were recorded by using linear probe and three different ultrasound machines namely Sonoscape, Lumify, and Hitachi. To make the analysis and annotation process easier, the original data in the form of videos were divided into individual 2D frames. The doctors decided to annotate every 10th frame, instead of each one, to reduce the workload while still being able to focus on the frames that were more likely to contain important features or relevant changes. They used the open-source annotation tool CVAT to annotate the frames in the dataset. The next step involved converting the multi-polygon vector annotation masks that were generated using CVAT into raster binary masks that solely focused on the signs of interest. By converting the annotations into raster binary masks, each pixel within the focused sign region was assigned a value of either 0 or 1, indicating the absence or presence of the sign [29].

Transformation of LUS videos into applicable images with corresponding segmentation mask was carried out by doctors, as well as our tutor, Ing. Maroš Hliboký, who provided us with the training dataset. The training dataset consisted of two folders, one containing images and the other containing corresponding segmentation

masks. In addition, a JSON file was given to us which contains all the necessary information about each image in the dataset such as the path to the image, the path to all segmentation masks, and details about any artifacts present in the image. The whole dataset consisted of 5,896 images that were obtained by slicing 558 videos. A-lines were only identifiable in a subset of 39 videos, which were sliced into 1,074 individual images. B-lines were present in an even smaller subset of only 33 videos, resulting in a total of 346 images obtained by slicing these videos. More information about our training data can be seen in Table 5 – 1.

Probe type	Videos	Frames	Videos	Frames
Hitachi	15	395	17	154
Sonoscape	6	231	3	27
Lumify	18	448	13	165
	A-lines		B-lines	

Tab. 5 – 1 Training data distribution

To use images of A-lines and B-lines for training GANs, first, we had to pre-process them. The pre-processing stage consisted of several sequential steps. Firstly, we extracted only images that contained A-line or B-line artifacts from the dataset. As a single image may contain multiple A-lines or B-lines artifacts, we concatenated all corresponding binary masks belonging to the same image. After that, we normalized the pixel values of all images to be within the range of [-1, 1]. This normalization helps the activation functions within the discriminator to work effectively, thereby enhancing feature extraction capabilities. In the final step of the process, we resized all the images to a uniform size to ensure consistency during training. To achieve this, we first calculated the average height and width for images of A-lines and B-lines. The average size for images of A-lines was 450x271, and for images of B-lines was 447x265. Since we intended to use convolutional layers in our models that halve the size of the input, we decided that the final size should be 512x256. This size is close to the average size, and we can halve it without having to round the resolution

to the nearest integer. During training of GANs, we used only horizontal flip to augment images in our dataset. This expands the training data with variations, allowing the model to learn from a broader range of features and reducing the risk of overfitting. We avoided other cropping or rotation techniques because they could deform lung artifacts. For instance, rotation by 90 degree could transform A-lines into B-lines and vice versa, which would be undesirable. In Figure 5 – 1 and Figure 5 – 2 you can see example of A-line and B-line with corresponding mask.

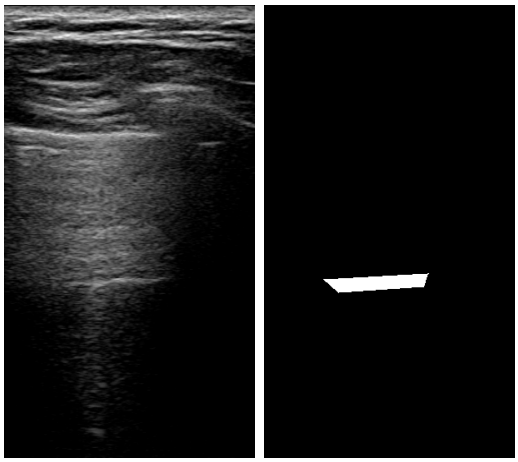

Figure 5 – 1 Example of A-line with corresponding mask

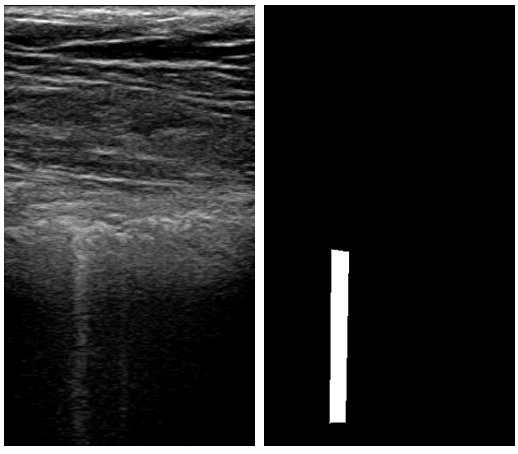

Figure 5 – 2 Example of B-line with corresponding mask

5.2 Deep Convolutional Generative Adversarial Network

Deep Convolutional Generative Adversarial Network or DCGAN for short, is a type of GANs that is specifically designed to generate realistic images. DCGAN uses deep convolutional neural network (CNN) as the base architecture for both the generator and discriminator. The purpose of these CNNs is to capture spatial dependencies and local patterns within the image. By using this approach, DCGAN can generate high-resolution images that have fine-grained details [30]. While DCGAN leverages the power of CNN, it incorporates specific modifications to optimize image generation. Here are some key ways DCGAN adapts CNN architecture.

Traditional CNN often use pooling layers such as max or average pooling to downsample the input image and extract higher-level features [31]. However, this reduces spatial resolution, which is not ideal for image-generation tasks. DCGAN, on the other hand, eliminates pooling layers entirely and instead uses strided transposed convolutions in the generator network and strided convolutions in the discriminator network. This enables DCGAN to learn its own optimal methods for downsampling and upsampling during the training process [30]. In DCGAN, batch normalization is applied to both the generator and discriminator networks. By normalizing the activations at each layer, batch normalization ensures that the networks are more robust to variations in input data and facilitates faster and more stable training [32]. Unlike traditional CNN architectures, which often include fully connected layers at the end of the network for classification tasks, DCGAN does not utilize fully connected layers [31]. Instead, the final layers of both the generator and discriminator networks consist of convolutional or transposed convolutional layers followed by an activation function. This architectural choice eliminates the need for flattening the feature maps before passing them to fully connected layers, which reduces the number of parameters and helps preserve spatial information. By avoiding fully connected layers, DCGAN can generate images with fine-grained details, making them more suitable for generative tasks where preserving spatial structure is crucial[30].

So, DCGAN is a type of GANs that employs a modified version of CNN for both the generator and discriminator. During training, it utilizes the vanilla GAN objective function to generate fake images, the mathematical expression of which can be observed in Equation 3.1.

5.2.1 DCGAN: Generator architecture

We followed the guidelines of the DCGAN authors to create the generator with the following architecture. The generator is made up of 8 blocks, each of which gradually upsamples an N-dimensional noise vector that comes from a gaussian distribution into a high-resolution image. All blocks, except the last one, consist of a transposed convolutional layer followed by a Batch Normalization and then a ReLU activation function. The last block only consists of a transposed convolutional layer and a Tanh activation function to scale the output image pixels into the range of [-1, 1]. To generate images with a resolution of 512x256 pixels, specific kernel sizes and strides were utilized throughout the network. The first transposed layer is different from the rest, as it employs a 4x2 kernel size with a stride of 1 and no padding. The subsequent transposed layers use a 4x4 kernel size with a stride of 2 and padding of 1. The generator architecture can be seen in Figure 5 – 3.

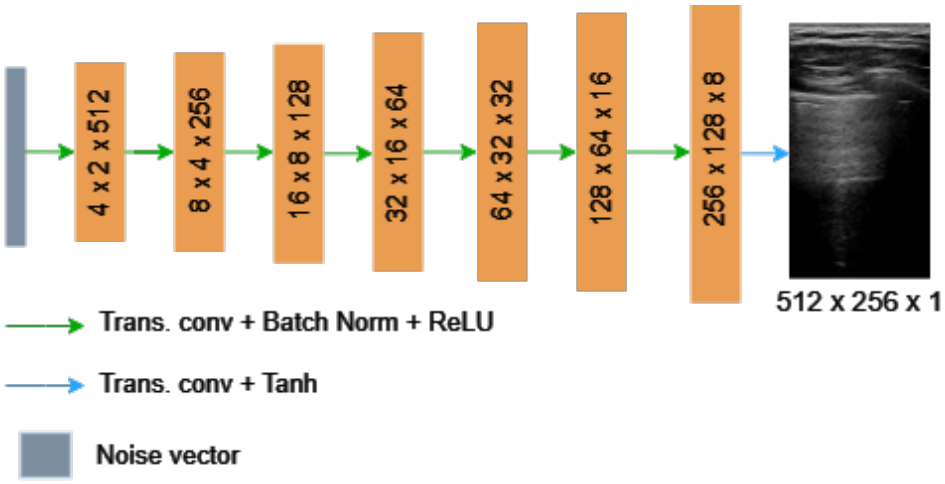

Figure 5 – 3 DCGAN: Generator architecture

5.2.2 DCGAN: Discriminator architecture

The discriminator has a similar architecture to the generator, but it utilizes convolutional layers instead of transposed convolutional layers. In each block, except the last one, there is a convolutional layer followed by a Batch Normalization and then a Leaky ReLU activation function with a slope of 0.2. The final block uses only a convolutional layer and a Sigmoid activation function, which scales the output value between 0 and 1, representing the probability of the input image being real. The first seven convolutional layers use a kernel size of 4x4 with a stride of 2 and padding of 1. The last layer uses a kernel size of 4x2 with a stride of 1 and padding of 0. The discriminator takes as input a batch of real or fake grayscale images with a shape of 512x256 pixels. The output is a single value that represents the probability of the input image being real. The discriminator architecture is visualized in Figure 5 – 4.

Figure 5 – 4 DCGAN: Discriminator architecture

5.3 Wasserstein GAN with Gradient Penalty

Wasserstein GAN with Gradient Penalty, commonly known as WGAN-GP, is a type of GANs that was developed by Martin Arjovsky and his team in 2017. The main goal of WGAN-GP is to enhance the stability of training the generative model. Although it has a dense mathematical motivation, in practice, it requires only a few

minor modifications to the established standard DCGAN [33].

The first modification is that unlike DCGAN, which optimize the Jensen-Shannon divergence or the Kullback-Leibler divergence, WGAN-GP aims to minimize the Wasserstein distance between the distribution of generated samples and the distribution of real samples. By minimizing the Wasserstein distance, it aims to produce more realistic samples while preventing mode collapse and training instability [33]. The Wasserstein distance, which is also known as the Earth Mover's Distance or Kantorovich-Rubinstein metric, is a way to measure the dissimilarity between two probability distributions. To understand this concept, imagine two probability distributions represented by piles of dirt or sand. Each pile represents the probability distribution of some feature like intensity values in an image. The Wasserstein distance between these two distributions represents the minimum amount of "work" or "effort" required to transform one pile into the other. The "work" is quantified by the amount of dirt moved times the moving distance [16] [34].

The second modification is that the architecture of WGAN-GP is slightly different from that of DCGAN. Rather than using a discriminator to classify or predict the probability of generated images as real or fake, it replaces the discriminator model with a critic model that scores the realness or fakeness of an image. This change is motivated by a mathematical argument that training the generator should seek a minimization of the distance between the distribution of the data observed in the training dataset and the distribution observed in generated examples. To achieve effective and stable training, it is crucial to ensure that the critic network enforces 1-Lipschitz continuity [33]. A function is Lipschitz continuous if small changes in its input lead to relatively small changes in its output. Mathematically, there exists a constant K such that the absolute value of the change in the function's output is always less than or equal to K times the absolute value of the change in the input [35]. To ensure Lipschitz continuity, a regularization technique called gradient penalty is used. The idea behind the gradient penalty is that functions are 1-Lipschitz if their

gradients have a norm (magnitude) of at most 1 everywhere. The gradient penalty adds a term to the loss function of the GAN that penalizes large gradients. To compute the gradient penalty, interpolated samples need to be generated by mixing real and generated data. Then, the gradients of the critic's output with respect to these interpolated samples are calculated. Finally, the penalty is computed by taking the squared difference between the gradient norms and a target norm 1 and averaging over all samples [33].

The objective function used in WGAN-GP is shown below:

$$\min_G \max_C L = \underbrace{\underset{z\sim P_{noise}}{E}[Critic(G(z))] - \underset{x\sim P_r}{E}[Critic(x)]}_{\text{Original critic loss}} + \underbrace{\lambda \underset{\hat{x}\sim P_{\hat{x}}}{E}[(\|\nabla_{\hat{x}} Critic(\hat{x})\|_2 - 1)^2]}_{\text{Gradient penalty}} \tag{5.1}$$

where, the first half $E[Critic(G(z))] - E[Critic(x)]$ estimates Earth Mover's distance between generated and real images, and the second half $\lambda \underset{\hat{x}\sim P_{\hat{x}}}{E}[(\|\nabla_{\hat{x}} critic(\hat{x})\|_2 - 1)^2]$ is the regularization term that tries to make the critic 1-L continuous so that the loss function is continuous and differentiable. λ is a hyperparameter that controls the weight of the gradient penalty term relative to the Wasserstein distance term. P_{noise} represents distribution of random noise, P_r represents distribution of real data, and $P_{\hat{x}}$ represents interpolated samples [33].

5.3.1 WGAN-GP: Generator architecture

We conducted two types of experiments using WGAN-GP. The first type focused on unconditional generating, where the input to the generator is only a noise vector. In this type of experiments, the generator has the same architecture as DCGAN and can be seen in Figure 5 – 3. The second type of experiments focused on conditional generating. In this type of experiments, a noise vector and binary mask are used as input to the generator. Binary mask guide the generator to create A-lines and B-lines at specific positions. To integrate a binary mask as input we used an encoder-decoder

architecture for the generator, which can be seen in the Figure 5 – 5. The encoder part encodes a binary mask into a latent vector, which is then multiplied by random noise and transformed by the decoder part to the output image. The encoder part consists of a convolutional layer followed by Batch Normalization and Leaky ReLU with a slope of 0.2. The decoder part consists of a transposed convolutional layer followed by Batch Normalization and ReLU. After the last transposed convolutional layer, the Tanh function is used, to scale pixel values into the range [-1, 1]. All convolutional layers (except the last) use a 4x4 kernel with stride 2 and padding 1. The last layer uses a 4x2 kernel with stride 1 and no padding. Similarly, all transposed convolutional layers (except the first) use a 4x4 kernel with stride 2 and padding 1. The first transposed layer uses a 4x2 kernel with stride 1 and no padding.

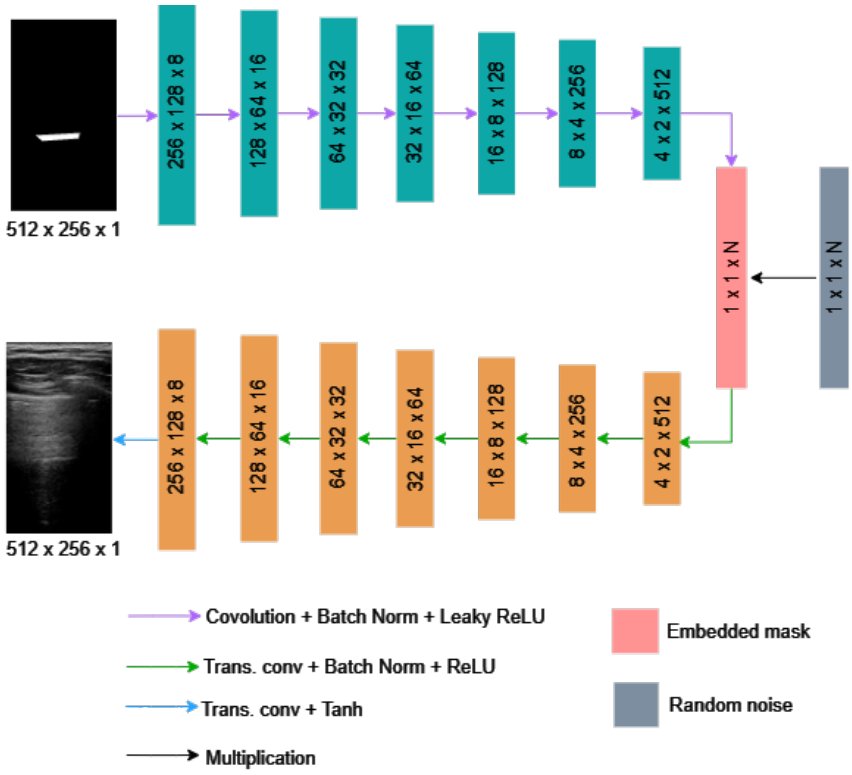

Figure 5 – 5 WGAN-GP: Generator architecture

5.3.2 WGAN-GP: Critic architecture

For both types of WGAN-GP generators we employed the same critic architecture. The critic's architecture is similar to DCGAN dicriminator's architecture, but with some minor modifications. We replaced Batch Normalization with Instance Normalization because Batch Normalization introduces batch correlation, whereas the gradient penalty should be imposed independently on different samples. We also did not use the Sigmoid function in the final block because, unlike DCGAN which uses the Sigmoid function to output probabilities between 0 and 1, WGAN-GP focuses on distance using the Wasserstein loss. Therefore, the critic's output only needs to be a real number. Critic architecture can be seen in Figure 5 – 6.

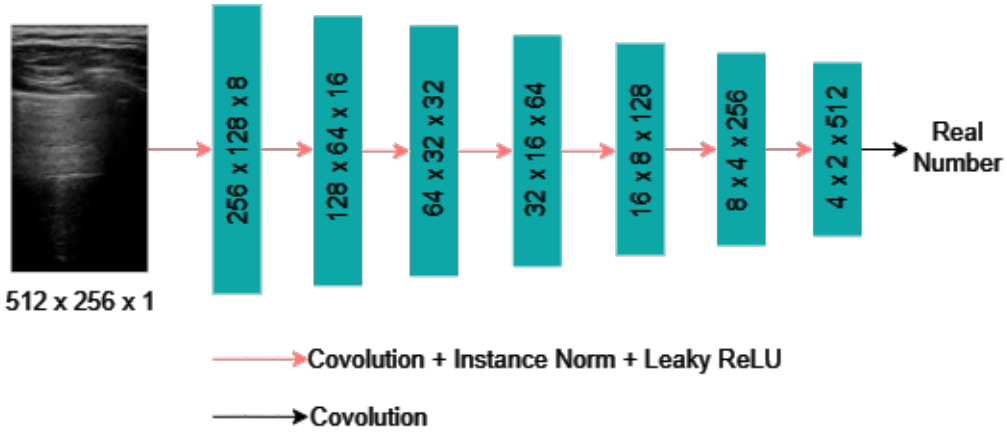

Figure 5 – 6 WGAN-GP: Critic architecture

5.4 Pix2Pix

Pix2Pix is a groundbreaking deep learning model that has revolutionized the field of image-to-image translation. It was introduced by Isola et al. in 2017 and offers a flexible framework for generating realistic images. This is achieved by learning the mapping between input and output images in a supervised manner. Unlike traditional GANs that generate images from random noise, Pix2Pix takes an input image and produces a corresponding output image. This conditional setting en-

ables Pix2Pix to learn complex mappings between different image domains, such as converting a binary segmentation mask to a corresponding LUS image [36].

The Pix2Pix generator architecture is a variant of the U-Net design. Both networks share an encoder-decoder structure with skip connections, but with some key differences. Pix2Pix uses only one convolutional layer before downsampling, while the original U-Net used two. Additionally, U-Net scales down the input image to a maximum resolution of around 32x32, whereas the Pix2Pix generator network performs downsampling all the way to 1x1. To introduce randomness during image generation, Pix2Pix employs dropout layers in the generator network during both training and testing, eliminating the need for random noise as input [36] [37].

The Pix2Pix distinguishes itself from standard GANs by utilizing a PatchGAN discriminator instead of a deep CNN for image classification. Unlike the standard discriminator, which evaluates the entire image as real or fake, the PatchGAN discriminator operates at a local level. It analyzes individual N x N patches within the image to determine their authenticity. This approach enables the discriminator to capture the textures and styles of an image, which the standard discriminator cannot. The patch size in PatchGAN can be customized for each task, but the final output is an average of all the responses of the patches considered. The PatchGAN discriminator has several advantages over the standard GAN discriminator, including fewer training parameters, faster speed, and the ability to be applied to arbitrarily large images [36] [38].

Similar to standard GANs models, the Pix2Pix discriminator aims to minimize the negative log-likelihood of correctly classifying real and fake images, conditioned on the corresponding source image. The mathematical expresion of the objective function can be seen in Equation 3.2. However, the discriminator model trains much faster than the generator, which can lead to an imbalance during a training. To address this, the loss of the discriminator is halved to slow down its training process. The generator model is trained using both the adversarial loss for the discriminator

model and the L1 or mean absolute pixel difference between the generated translation of the source image and the expected target image. The adversarial loss influences whether the generator model can output images that are plausible in the target domain. The L1 loss regularizes the generator model to output images that are a plausible translation of the source image. These two loss functions are combined into a single composite objective function, which is then used to update the generator model [36].

Generator objective function can be seen in equation below.

$$L = AdversarialLoss + \lambda \underbrace{E_{x,y}[\|y - G(x)\|_1]}_{\text{L1 loss}} \tag{5.2}$$

where y is target image, and λ is a hyperparameter to adjust the balance between the adversarial loss and the L1 distance [36].

5.4.1 Pix2Pix: Generator architecture

We have developed a generator based on the guidelines provided by the Pix2Pix authors. The generator architecture comprises 8 downsampling and 8 upsampling blocks, connected by skip connections. Each downsampling block consists of a convolutional layer followed by Batch Normalization and Leaky ReLu with a slope of 0.2. With the exception of the last block, all upsampling blocks consist of transposed convolution, followed by Batch Normalization and ReLu activation. The last block comprises a transposed convolution and Tanh activation to scale pixel values of the generated image into the range of [-1, 1].

The 512x256 input mask is transformed into a 1x1 vector by using a kernel size of 4x4 with a stride of 2 and padding of 1 in the first seven downsampling blocks, and a kernel size of 4x2 with a stride of 1 and padding of 0 in the last downsampling block. The embedded mask is then converted into a generated image of size 512x256 by employing a kernel size of 4x2 with a stride of 1 and padding of 0 in the first upsampling block, followed by a kernel size of 4x4 with a stride of 2 and padding

of 1 in the subsequent upsampling blocks. Dropout was introduced in the second, third, and fourth upsampling blocks to introduce randomness into the generation process. The generator architecture can be seen in Figure 5 – 7.

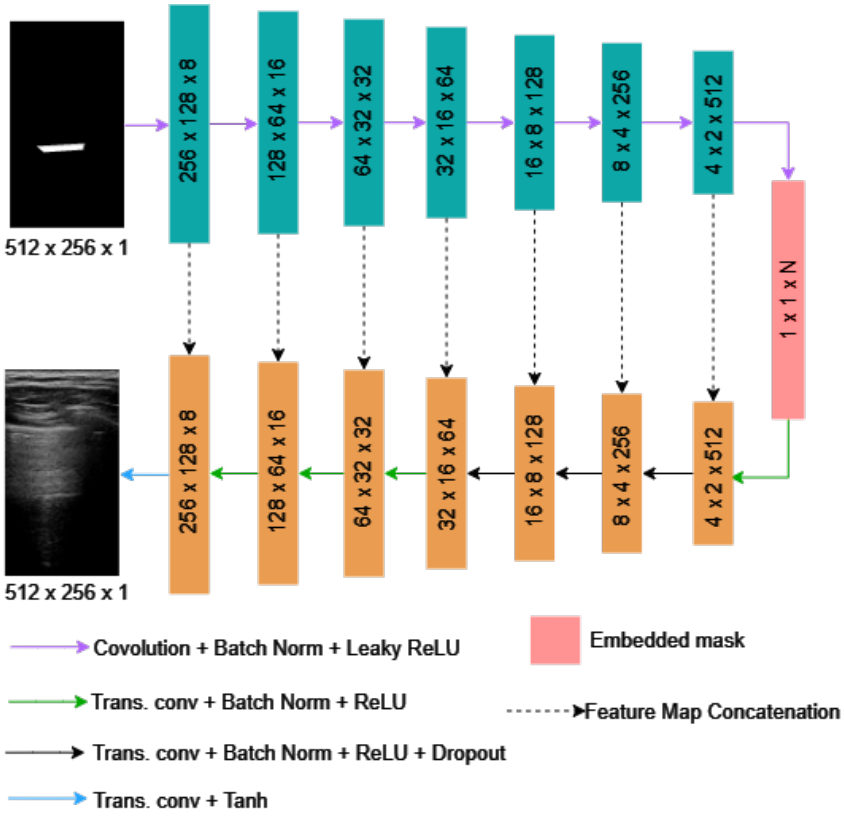

Figure 5 – 7 Pix2Pix: Generator architecture

5.4.2 Pix2Pix: Discriminator architecture

Our PatchGAN discriminator architecture consists of five downsampling blocks. Each block, except for the last one, consists of a convolutional layer, followed by Batch Normalization, and Leaky ReLU activation with a slope of 0.2. The last block only consists of convolutional layer and Sigmoid activation function. The first three convolutional layers have a kernel size of 4x4 with a stride of 2 and padding of 1, and the rest have a kernel size of 4x4 with a stride of 1 and padding of 1. The discriminator takes a pair of an image and a segmentation mask as input, and its output is a ”grid” of size 62x30, where each square contains a number that evaluates

140x70 patch of the original image to determine if it is real or fake. The discriminator architecture can be seen in the Figure 5 – 8.

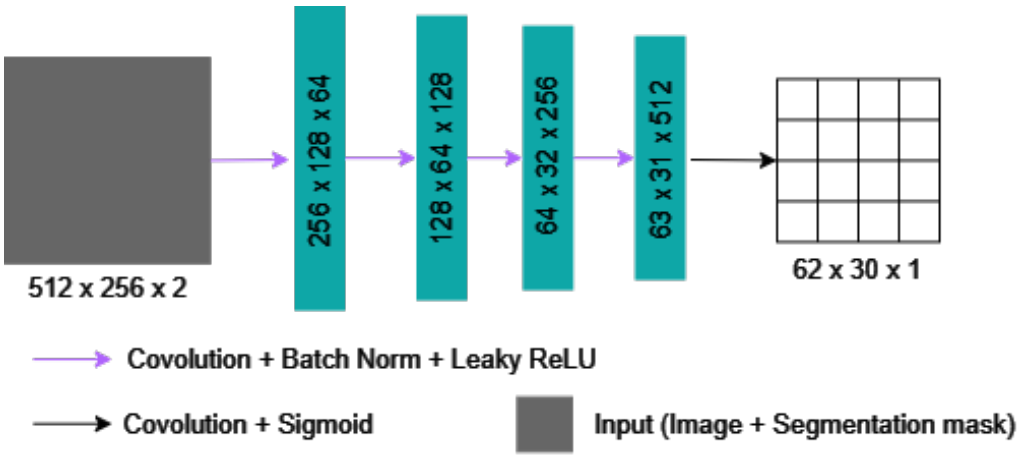

Figure 5 – 8 Pix2Pix: Discriminator architecture

6 Experiments and results

In this section, we will provide all the information about our experiments. The main objective of our experiments is to train a GANs to generate images of A-lines and B-lines. We used the three described GANs types, which were detailed in the previous section. Our experiments can be categorized into two types. In the first type of experiments, we used unconditional synthesis, which means that only noise drawn from gaussian distribution was used as input for the generator. For this type, we employed DCGAN and WGAN-GP. On the other hand, in the second type of experiments, we used conditional synthesis, where the generator used a binary mask to guide the whole generating process. For this, we used WGAN-GP and Pix2Pix. How binary mask looks like can be seen in Figure 5 – 1. We used all 1,074 images of A-lines exclusively to train our GAN models in all experiments. This decision stemmed from the fact that we possessed three times as many images of A-lines as B-lines. Once we determined the optimal model configuration for images of A-lines, we subsequently trained the same models using all 346 images of B-lines.

To assess the quality of generated images across all experiments, we employed the FID metric, which is explained in Subsection 3.4.3. Since image generation by GANs inherently involves a degree of randomness, we adopted a specific evaluation strategy. For each evaluation, we generated a set of images identical in size to the training dataset. We then computed the FID score between these generated images and the real training images. Then we repeated this process five times and calculated average FID score. The smaller the FID score, the more similar the generated data should be compared to real data. The FID score evaluates the similarity between generated images and real ones. However, it doesn't specifically indicate whether the generated images contain A-lines or B-lines artifacts. To gain a deeper understanding of our GANs' ability to generate images with A-lines and B-lines artifacts, we generated 500 images from each of our final experiment. These images were then incorporated into the original training dataset and used to train ResNet-18 for classification.

6.1 Uncoditional DCGAN

The purpose of this experiment was to determine whether DCGAN was a suitable solution for our problem. During the first five trials, we encountered an issue known as mode collapse. This mean that the generator started producing identical images repeatedly. As a result, we didn't proceed with further trials and began searching for an alternative solution. You can see an example of what mode collapse looks like in Figure 6 – 1. The detailed architecture of the generator and discriminator networks employed in this experiment is described in Subsection 5.2.1 and Subsection 5.2.2. Visual representations of these architectures can be found in Figure 5 – 3 and Figure 5 – 4, respectively. During the training process, the Adam optimization algorithm was employed to update networks parameters, using the adversarial objective function defined in Equation 3.1. Adam optimizer was used with a learning rate of 0.0001, β_1 of 0, and β_2 of 0.9.

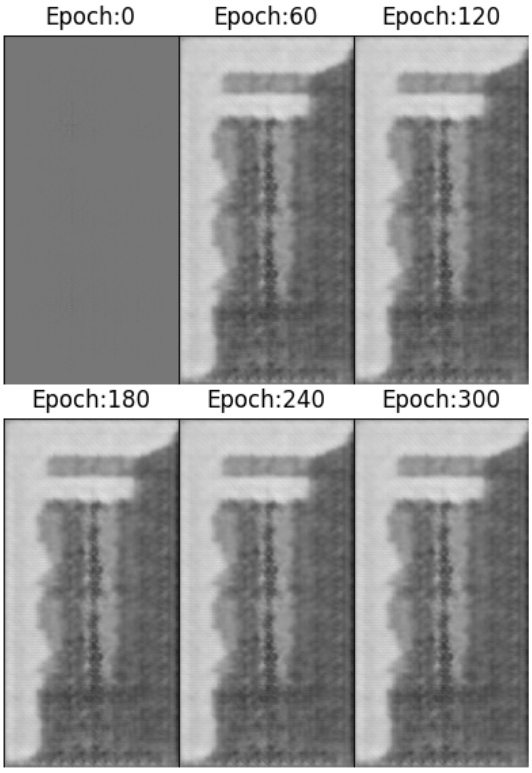

Figure 6 – 1 Generated images during training

6.2 Uncoditional WGAN-GP

In our next experiment, we used an unconditional WGAN-GP to address the issue of mode collapse in DCGAN. Similar to our previous experiment, we aimed to determine if this type of GANs is suitable for generating lung ultrasound images. The detailed architectures of the generator and discriminator networks employed in this experiment are described in Subsection 5.3.1 and Subsection 5.3.2. Visual representations of these architectures can be found in Figure 5 – 3 and Figure 5 – 6, respectively. During the training process, the Adam optimization algorithm was utilized to update network parameters, and the WGAN-GP objective function, which is defined in Equation 5.1. Adam optimizer was used with a learning rate of 0.0001, β_1 of 0, and β_2 of 0.9. It's important to note that the WGAN-GP objective function incorporates a hyperparameter λ, which controls the weight of the gradient penalty term. In these experiments, we adopted value of 10 for λ as recommended in the WGAN-GP paper. Additionally, it's worth noting that with the gradient penalty, we penalize the critic more than the generator, resulting in a slower learning rate for the critic. To maintain balance, we trained the critic five times more than the generator, as recommended by the authors of WGAN-GP.

Given the promising performance of WGAN-GP compared to DCGAN in our initial trials, we focused on optimizing the latent space representation by evaluating the impact of noise vector dimensionality on the quality of generated images. We conducted a series of seven sub-experiments, systematically varying the dimension of the gaussian noise input vector. Each sub-experiment was train for 300 epochs and evaluated with FID score.

Noise size	200	250	300	350	400	450	**500**
FID	226.34	232.21	214.70	223.25	273.2	245.11	**212.27**

Tab. 6 – 1 Results of sub-experiments for a noise size

The results presented in Table 6 – 1 indicate that the optimal size for our input vector

is 500. The objective of these 7 sub-experiments was just to determine the input vector size that generates the highest quality images. We acknowledge that training a GANs for only 300 epochs might not be sufficient, hence we conducted a final experiment. For our final experiment with unconditional WGAN-GP, we utilized the same architectures and hyperparameters as described at the beginning of this subsection, except for one modification. We now know that the input vector size must be 500. Every 50 epochs, the quality of generated images was evaluated using the FID score. This allowed us to monitor the model's performance and identify potential improvements throughout the training process.

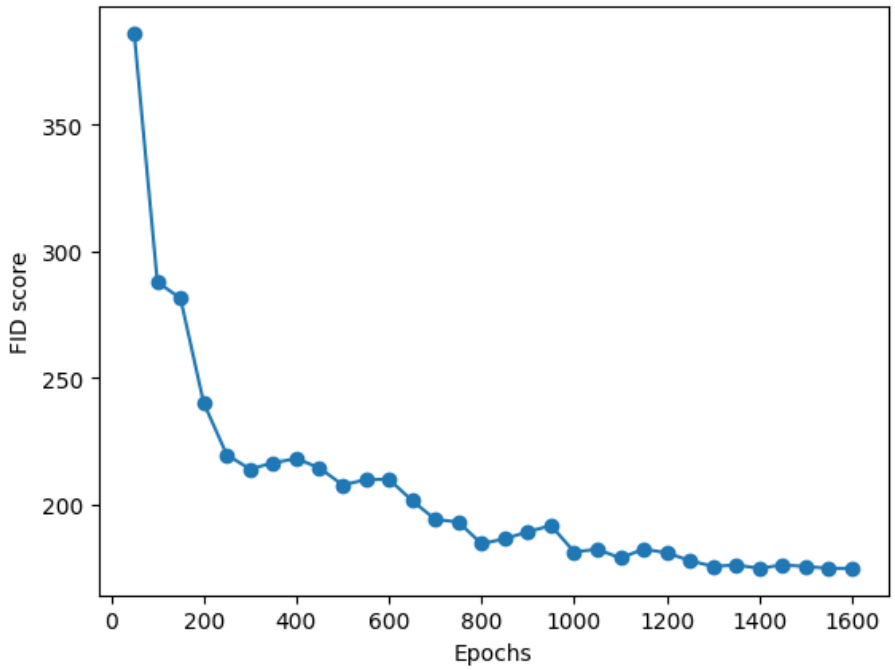

Figure 6 – 2 Progress in quality of generated images via WGAN-GP-NM (trained on A-lines)

Our model's progress over time is displayed in Figure 6 – 2. As expected, the FID score gradually decreases, indicating continuous improvement in the quality of generated images. This trend continued until approximately the 1300th epoch, at which point the FID score began to oscillate around a value of 175. We therefore didn't continue with training and used the generator parameters from the 1300th epoch to generate 500 images of A-lines. Figure 6 – 3 displays some examples of A-lines gen-

erated by our model. Upon examining the generated images, we can conclude that the generator attempts to synthesize images that could belong to the domain of lung ultrasound. However, the quality of the generated images is inferior compared to the originals. These images contain deformed areas, and the A-lines artifacts are of poor quality and hardly visible. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

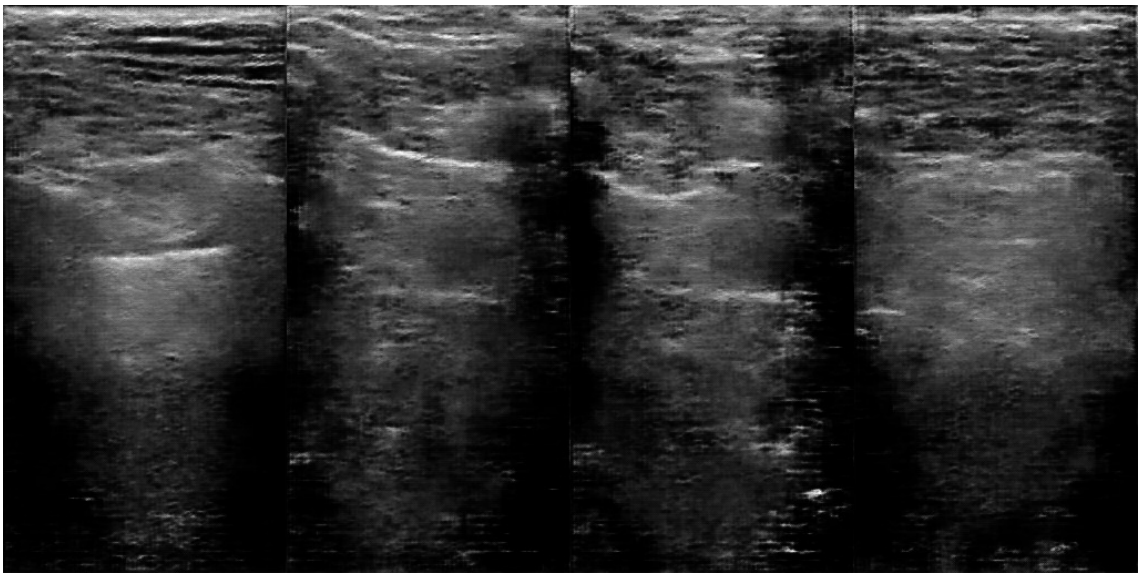

Figure 6 – 3 WGAN-GP-NM: generated A-lines

After training the model on images of A-lines in our final experiment, we swapped the training data for images of B-lines and trained it again. The model's progress over time is shown in Figure 6 – 4. Similarly to the previous experiment, the FID score gradually decreased until approximately the 1800th epoch. At this point, the FID score began to oscillate around a value of 167. As before, we decided to use the generator parameters from the 1800th epoch and generate 500 images of B-lines.

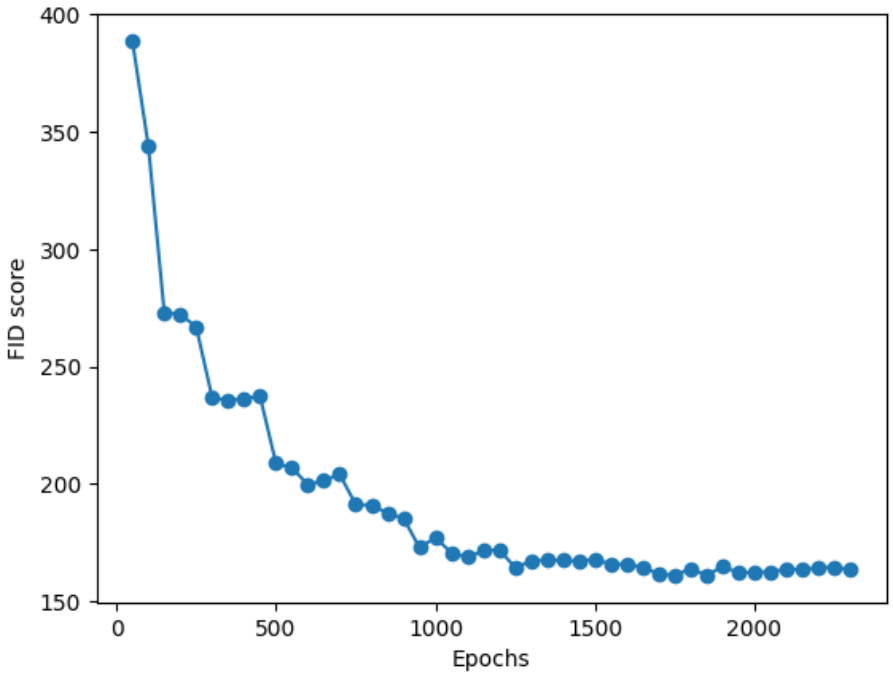

Figure 6 – 4 Progress in quality of generated images via WGAN-GP-NM (trained on B-lines)

Figure 6 – 5 illustrates representative examples of B-lines generated by our model. While the generated images appear to belong to the domain of lung ultrasound, their quality falls short of the originals. Examination of the generated images reveals the presence of deformed areas and the absence of B-lines artifacts, indicating the model's struggle in capturing these artifacts. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

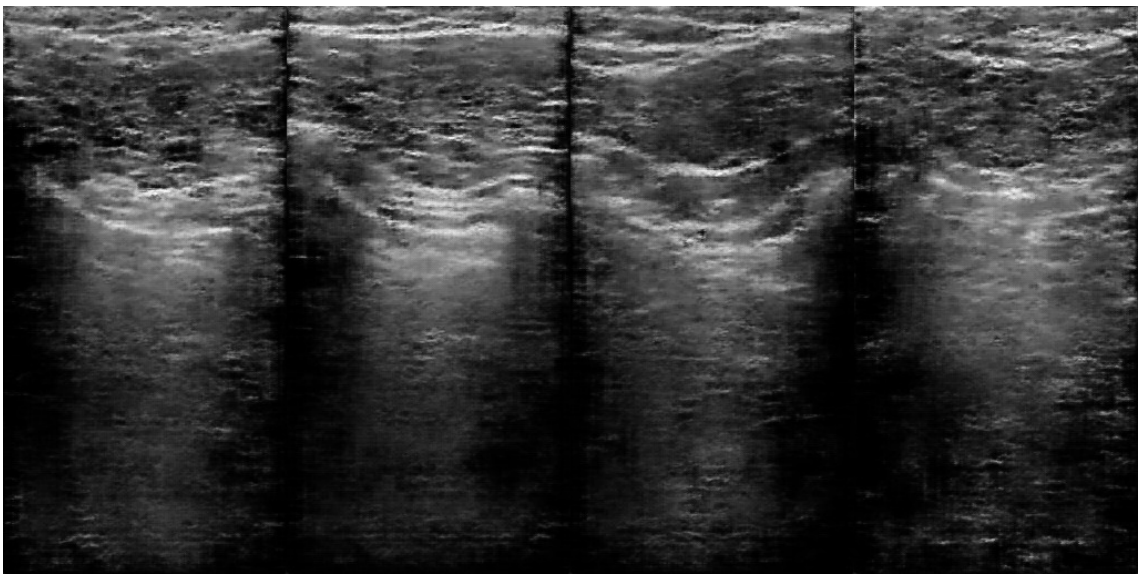

Figure 6 – 5 WGAN-GP-NM: generated B-lines

6.3 Conditional WGAN-GP

After observing the progress in image quality generated by WGAN-GP compared to DCGAN, we opted to enhance the generating process by incorporating a binary mask into the generator's input. This binary mask serves to guide the generator in producing lung artifacts at a specific position. However, integrating the binary mask as input required some modifications to the generator's architecture. The original architecture was designed to take a latent vector as input. However, the binary mask is a high-dimensional and structured input, which requires a different approach. For conditional synthesis with WGAN-GP, we employed two distinct generator architectures. The first architecture utilized an encoder-decoder structure as shown in Figure 5 – 5 and described in Subsection 5.3.1. The second architecture was a modified version incorporating skip connections within the encoder-decoder pathway. Notably, both architectures utilized the same discriminator, presented in Figure 5 – 6 and described in Subsection 5.3.2. Both generators take a binary mask as input and transform it with convolutional layers into a latent vector. This vector is then multiplied by noise drawn from a gaussian distribution. We followed the

findings from unconditional synthesis with WGAN-GP and set the noise vector's size to 500. This multiplication of the embedded mask with noise ensures diversity in the generated images. Subsequently, transpose convolutional layers are employed to convert the latent vector into a generated image.

6.3.1 Encoder-decoder type generator

During the training of the encoder-decoder type generator, we employed the Adam optimization algorithm to update networks parameters, and the WGAN-GP objective function, which is defined in Equation 5.1. The Adam optimization algorithm was employed with a learning rate of 0.0001, β_1 of 0, and β_2 of 0.9. To determine the optimal values for the hyperparameters λ and the number of critic iterations within our experiment, we conducted a series of controlled sub-experiments. Each sub-experiment maintained identical settings as described above, with the exception of varying λ and the number of critic iterations. All sub-experiments were trained for 300 epochs and subsequently evaluated using the FID score. By systematically exploring different combinations of these two hyperparameters, we achieved the lowest FID score when λ was set to 10 and the number of critic iterations was set to 5, aligning with the recommendations of the WGAN-GP authors. A number of critic iterations mean that we trained the critic 5 times more than the generator. Once we identified the optimal values, we conducted the final experiment, extending the training duration to more epochs. Every 50 epochs, the quality of generated images was evaluated using the FID score.

Figure 6 – 6 illustrates the training progress of the model when trained on images of A-lines. The FID score gradually declines over time, indicating a continuous improvement in the quality of generated images. This positive trend maintained until the 800th epoch, at which point the FID score began to oscillate around a value of 176. We therefore decided to take the generator parameters from the 800th epoch and generate 500 images of A-lines.

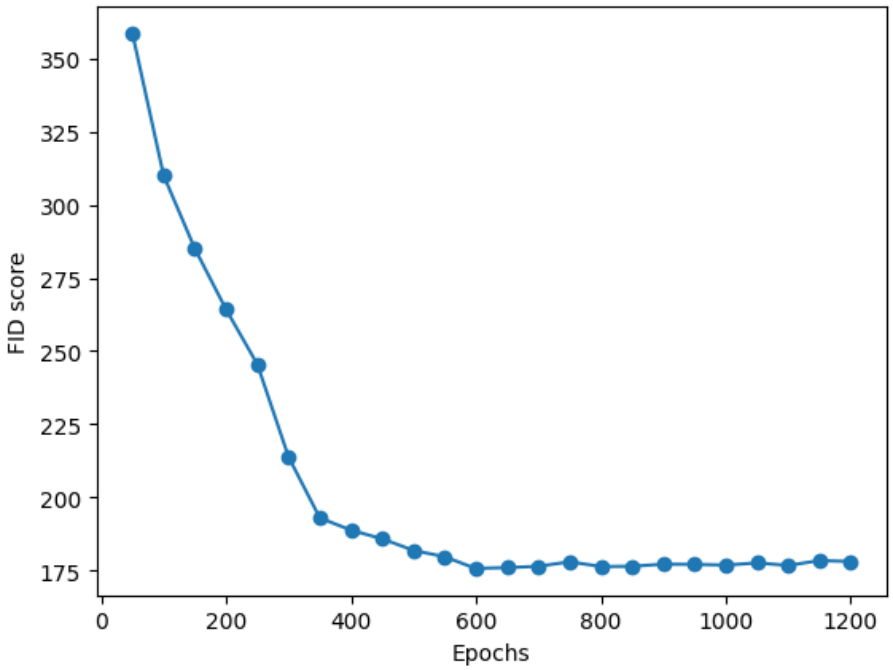

Figure 6 – 6 Progress in quality of generated images via WGAN-GP-ED (trained on A-lines)

Figure 6 – 7 illustrates representative examples of A-lines generated by our model. Visual inspection reveals an improvement in the quality of the generated images compared to those displayed in Figure 6 – 3. A-line artifacts appear to be of better quality and are easier to detect. Using binary masks as guidance appears to improve the generating process. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

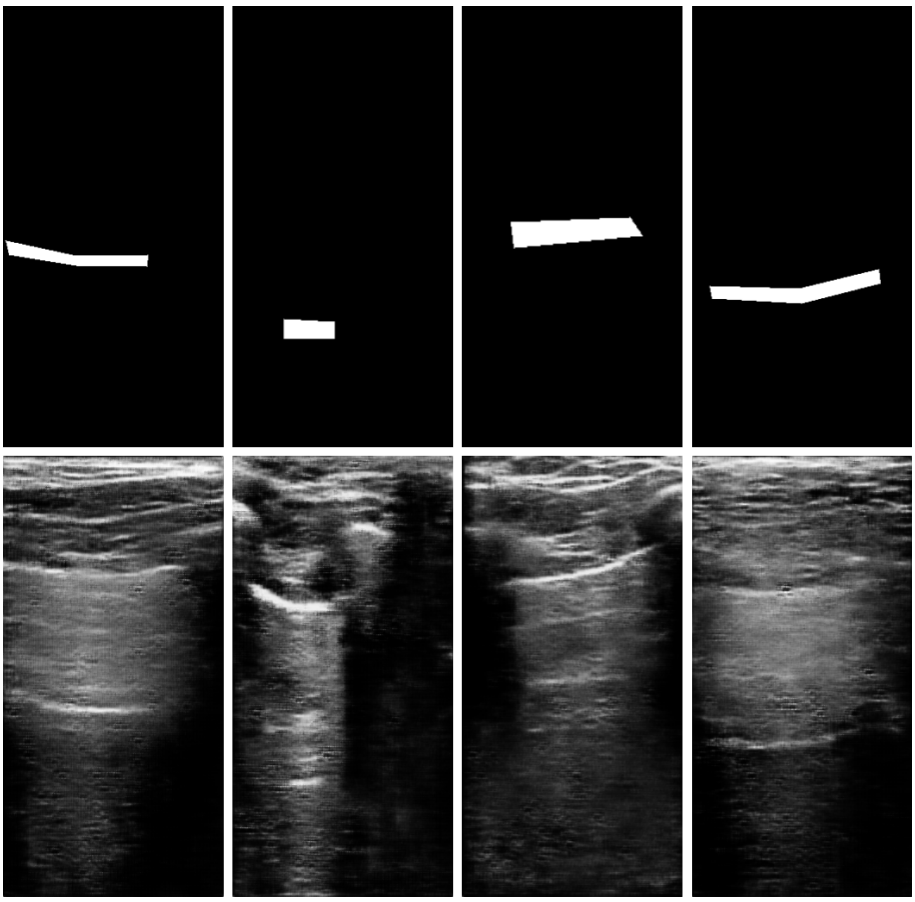

Figure 6 – 7 WGAN-GP-ED: generated A-lines, first row is input, second row is generated image

After training the model on images of A-lines in our final experiment, we swapped the training data for images of B-lines and trained it again. The model's progress over time is shown in Figure 6 – 8. Similarly to the experiment with images of A-lines, the FID score gradually decreased until approximately the 1250th epoch. At this point, the FID score began to oscillate around a value of 127. We therefore decided to take the generator parameters from the 1250th epoch and generate 500 images of B-lines.

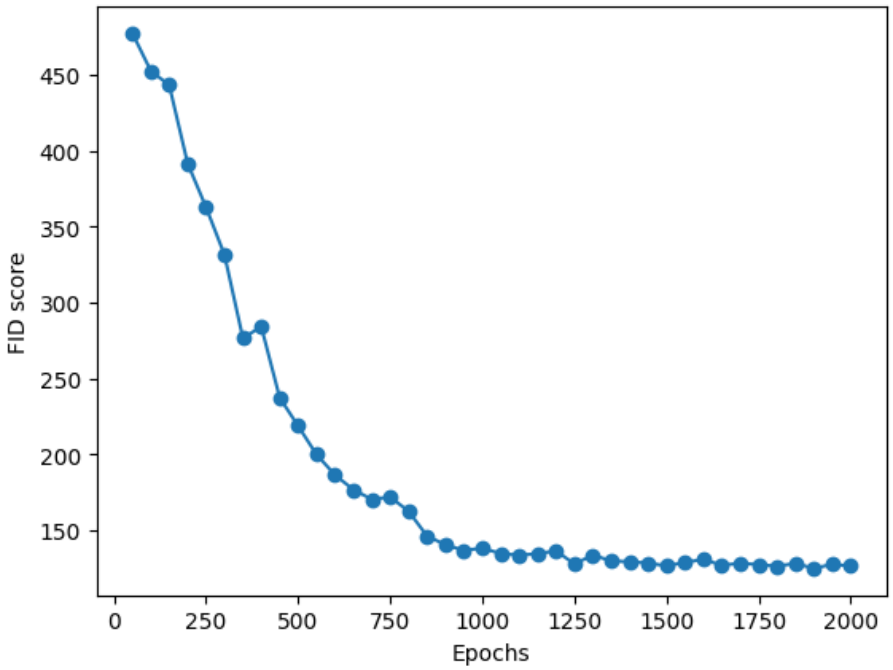

Figure 6 – 8 Progress in quality of generated images via WGAN-GP-ED (trained on B-lines)

Figure 6 – 9 illustrates representative examples of B-lines generated by our model. Similar to before, a visual inspection reveals an improvement in the quality of the generated images compared to those displayed in Figure 6 – 5. Significantly improvement can be observed in the generating of B-line artifacts, which were absent in the unconditional WGAN-GP experiment. Using the binary mask as guidance appears to improve the generating process. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

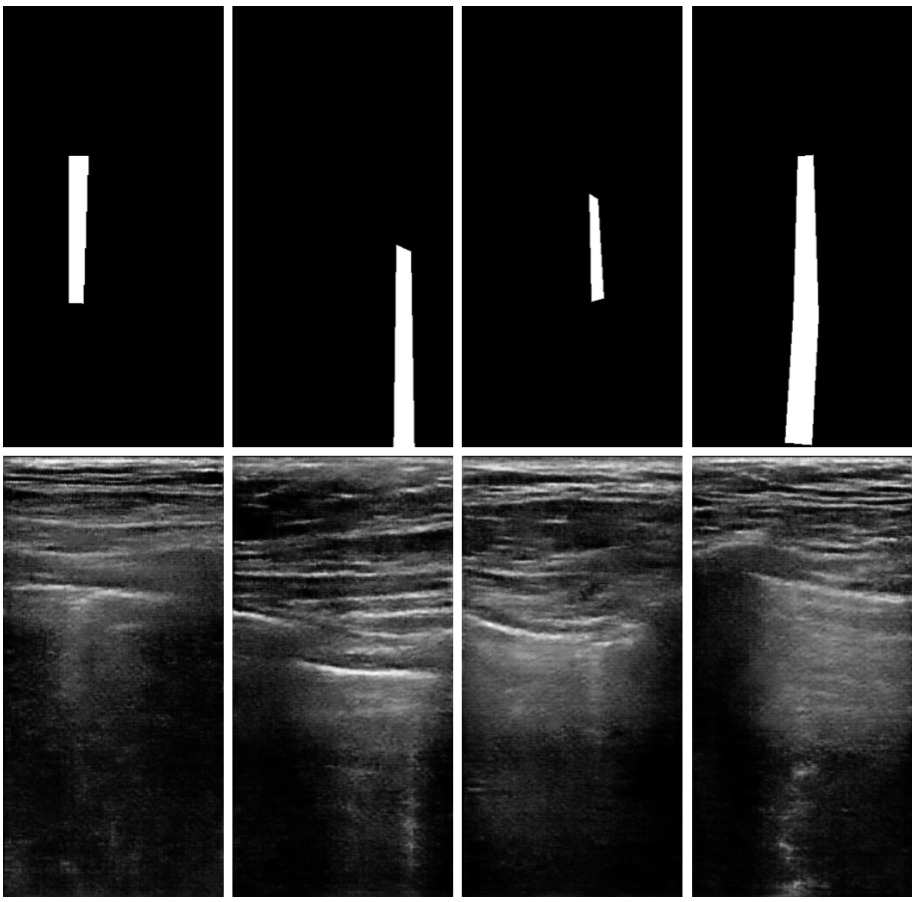

Figure 6 – 9 WGAN-GP-ED: generated B-lines, first row is input, second row is generated image

6.3.2 Encoder-decoder type generator with skip connection

Training the encoder-decoder with a skip connections generator employed the same procedures as the previous experiment. We maintained the same optimizer, learning rate, and hyperparameter tuning strategy (using sub-experiments) for finding the optimal λ and number of critic iterations. By systematically exploring different combinations of these two hyperparameters, we achieved the lowest FID score when λ was set to 15 and the number of critic iterations was set to 3. After identifying these optimal values, we conducted a final experiment with extended training for more epochs. Every 50 epochs, the quality of generated images was evaluated using the FID score.

Figure 6 – 10 illustrates the model's training progress when training on images of A-lines. The FID score gradually declines over time, indicating a continuous improvement in the quality of generated images. This positive trend persisted until the 800th epoch, at which point the FID score began to oscillate around a value of 152. We therefore decided to take the generator parameters from the 800th epoch and generate 500 images of A-lines.

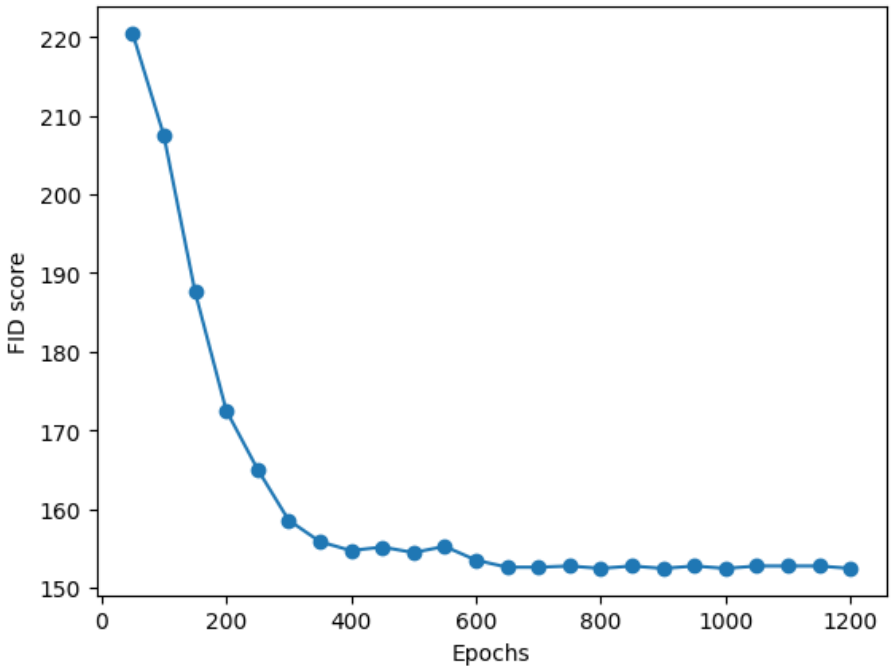

Figure 6 – 10 Progress in quality of generated images via WGAN-GP-SC (trained on A-lines)

Figure 6 – 11 presents the illustrative examples of A-lines generated by our model. The quality of the generated images appears very similar to the images in Figure 6 – 7. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

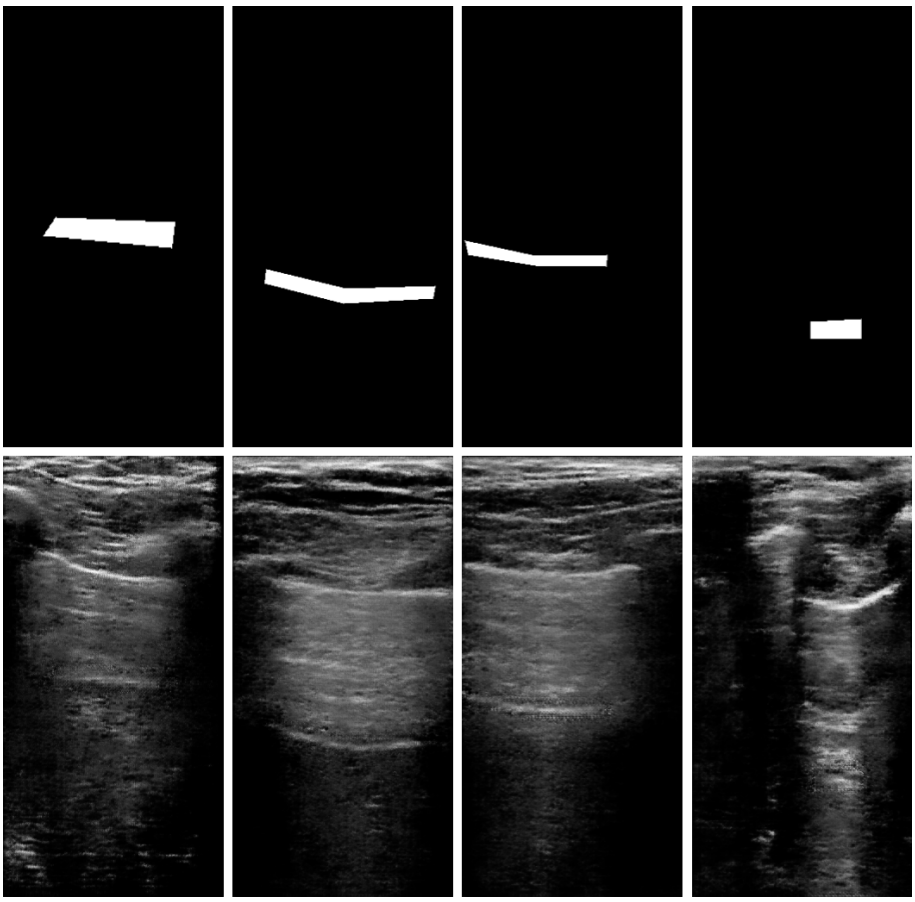

Figure 6 – 11 WGAN-GP-SC: generated A-lines, first row is input, second row is generated image

After training the model on images of A-lines in our final experiment, we swapped the training data for images of B-lines and trained it again. The model's training progress over time is illustrated in Figure 6 – 12. Similarly to the experiment with A-lines, the FID score gradually decreased until approximately the 550th epoch. At this point, our model reached the lowest FID score of 119. We therefore decided to take the generator parameters from the 550th epoch and generate 500 images of B-lines.

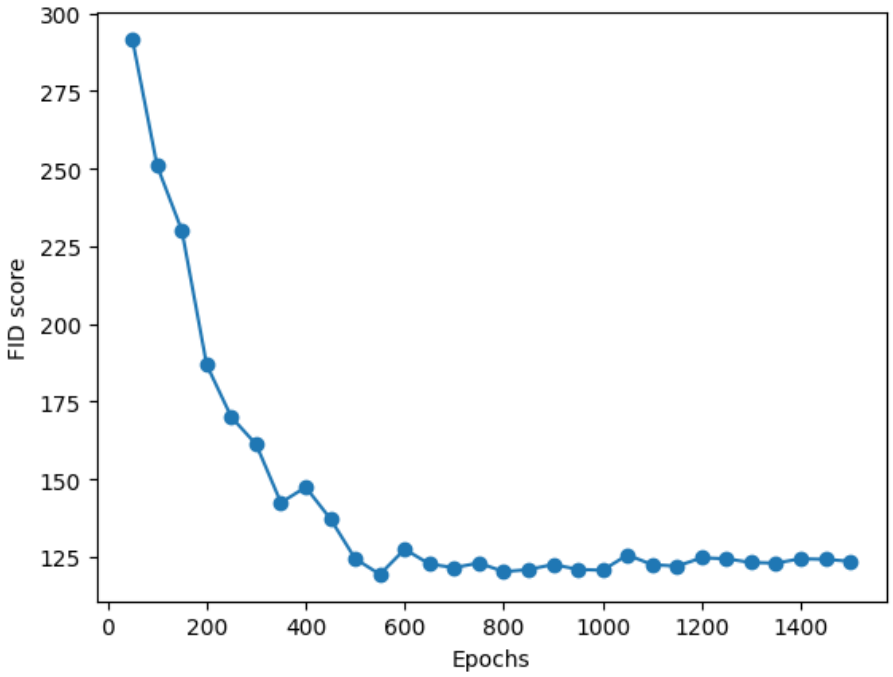

Figure 6 – 12 Progress in quality of generated images via WGAN-GP-SC (trained on B-lines)

Figure 6 – 13 presents the illustrative examples of B-lines generated by our model. The quality of the generated images appears very similar to the images in Figure 6 – 9. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

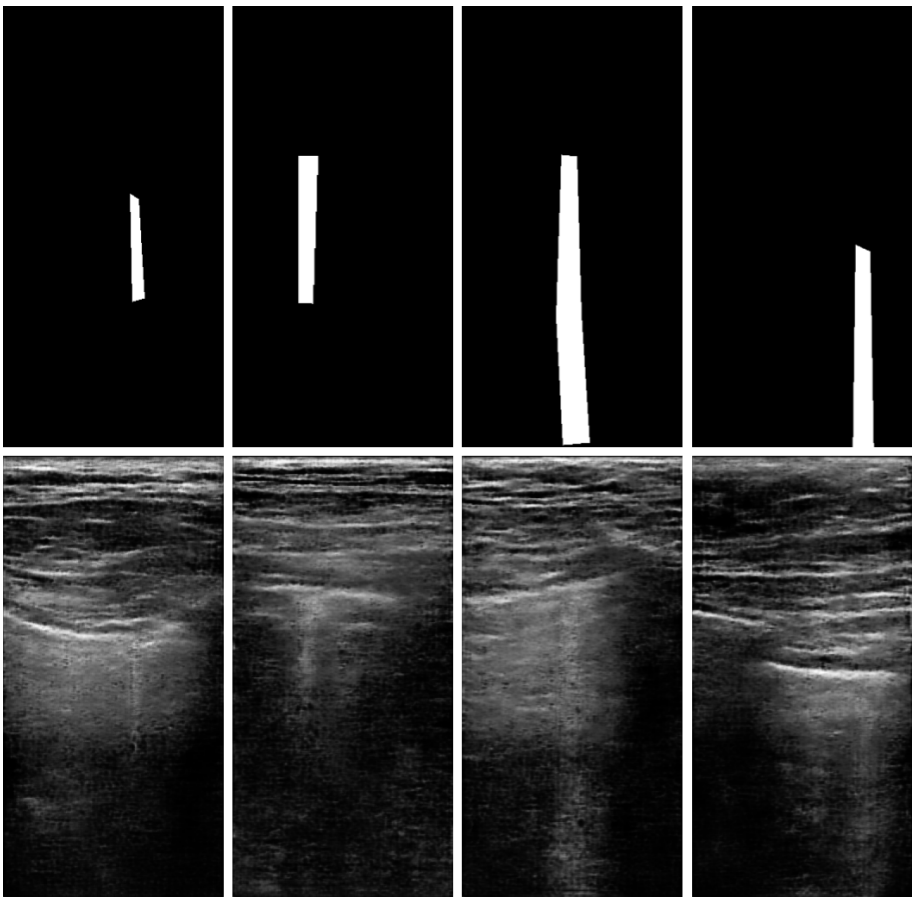

Figure 6 – 13 WGAN-GP-SC: generated B-lines, first row is input, second row is generated image

6.4 Conditional Pix2Pix

In our last experiment, we decided to explore the capabilities of Pix2Pix to address our challenge of generating lung ultrasound images. The detailed architectures of the generator and discriminator networks employed in this experiment are described in Subsection 5.4.1 and 5.4.2. Visual representations of these architectures can be found in Figures 5 – 7 and Figure 5 – 8, respectively. In this experiment, we used only a binary mask as input to the generator. Instead of a noise vector, we employed dropout layers in some generator blocks. The purpose of the dropout layers is to introduce randomness and prevent the model from simply memorizing the training

images. During the training process, the Adam optimization algorithm was utilized to update networks parameters, and the objective function defined in Equation 5.2. Adam optimizer was used with a learning rate of 0.0002, β_1 of 0.5, and β_2 of 0.999. As can be seen from Equation 5.2, the Pix2Pix objective function incorporates a hyperparameter λ, which determines the relative importance of the L1 loss compared to the adversarial loss for the generator during training. To determine the optimal value for λ, we conducted a series of controlled sub-experiments. All sub-experiments were trained for 300 epochs and then evaluated using the FID score metric.

λ	10	20	30	40	**50**
FID	196.45	198.23	172.83	180.59	**167.89**
λ	60	70	80	90	100
FID	182.31	184.09	190.84	182.01	207.57

Tab. 6 – 2 Results of sub-experiment for λ

The results of these sub-experiments, presented in Table 6 – 2, indicate that the optimal value for λ is 50. Once we found the optimal value for λ, we proceeded with the final experiment, where the model was trained for more epochs. Every 50 epochs, the quality of generated images was evaluated using the FID score.

Figure 6 – 14 illustrates the model's progress when training on images of A-lines. The FID score gradually declines over time, indicating a continuous improvement in the quality of generated images. This positive trend persisted until the 950th epoch, at which point the FID score began to oscillate around a value of 153. We therefore decided to take the generator parameters from the 950th epoch and generate 500 images of A-lines.

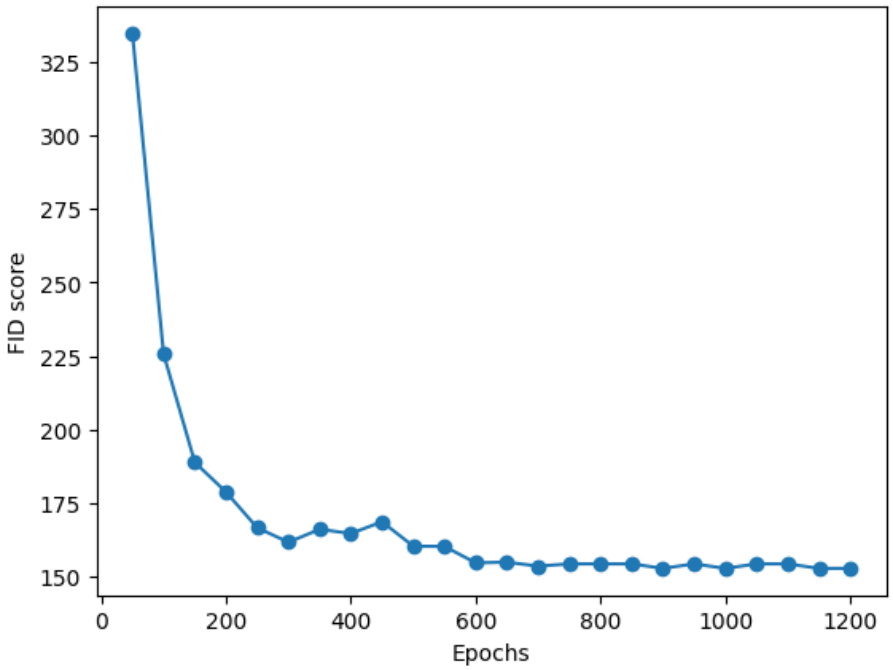

Figure 6 – 14 Progress in quality of generated images via Pix2Pix (trained on A-lines)

Figure 6 – 15 presents the illustrative examples of A-lines generated by our model. The quality of the generated images appears very similar to the images in Figure 6 – 7 and Figure 6 – 11. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Subsection 6.5.

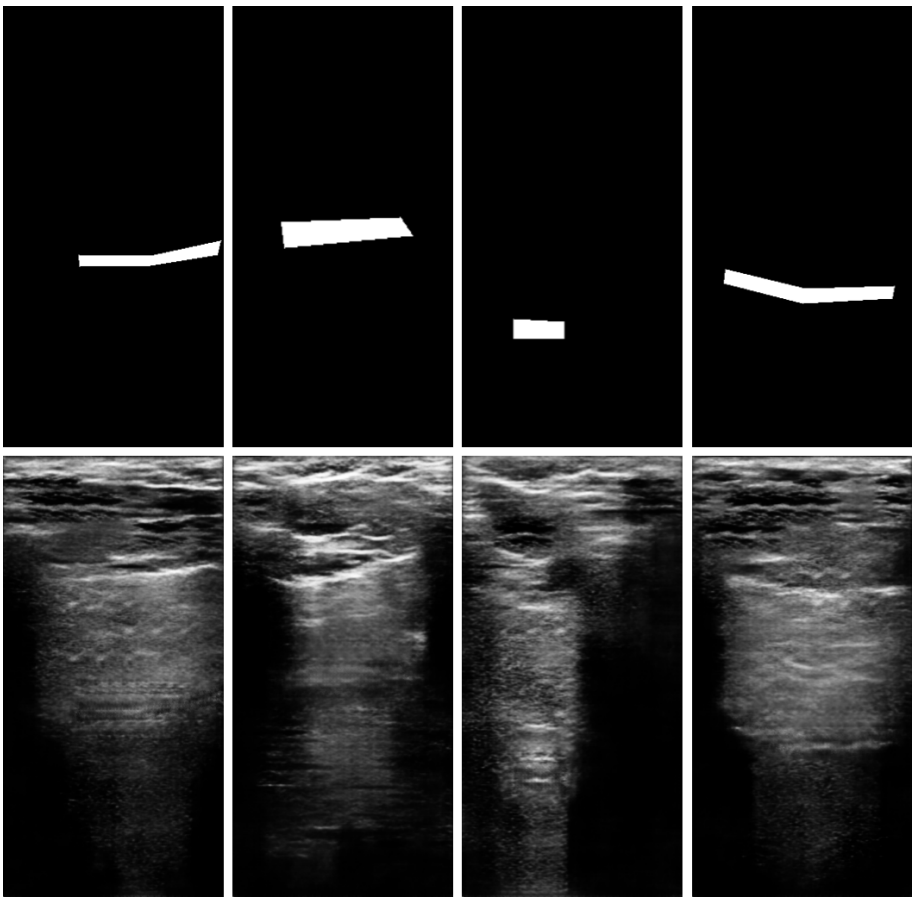

Figure 6 – 15 Pix2Pix: generated A-lines, first row is input, second row is generated image

After training the model on images of A-lines in our final experiment, we swapped the training data for images of B-lines and trained it again. The model's training progress over time is illustrated in Figure 6 – 16. Similarly to the experiment with A-lines, the FID score gradually decreased until approximately the 1200th epoch. At this point, our model reached the lowest FID of 155. We therefore decided to take the generator parameters from the 1200th epoch and generate 500 images of B-lines.

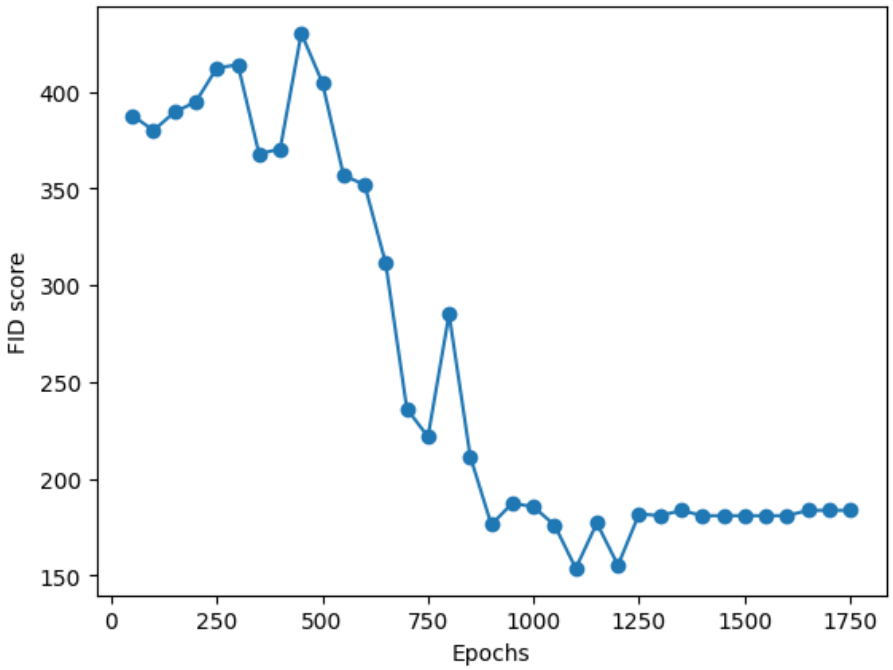

Figure 6 – 16 Progress in quality of generated images via Pix2Pix (trained on B-lines)

Figure 6 – 17 presents the illustrative examples of B-lines generated by our model. The quality of the generated images appears very similar to the images in Figure 6 – 9 and in Figure 6 – 13. One difference is that B-lines are generated with some kind of chessboard artifacts. To better assess the quality of the generated images, we employed them in training ResNet-18 to observe any potential improvement in model performance. The outcomes of this training are presented in Section 6.5.

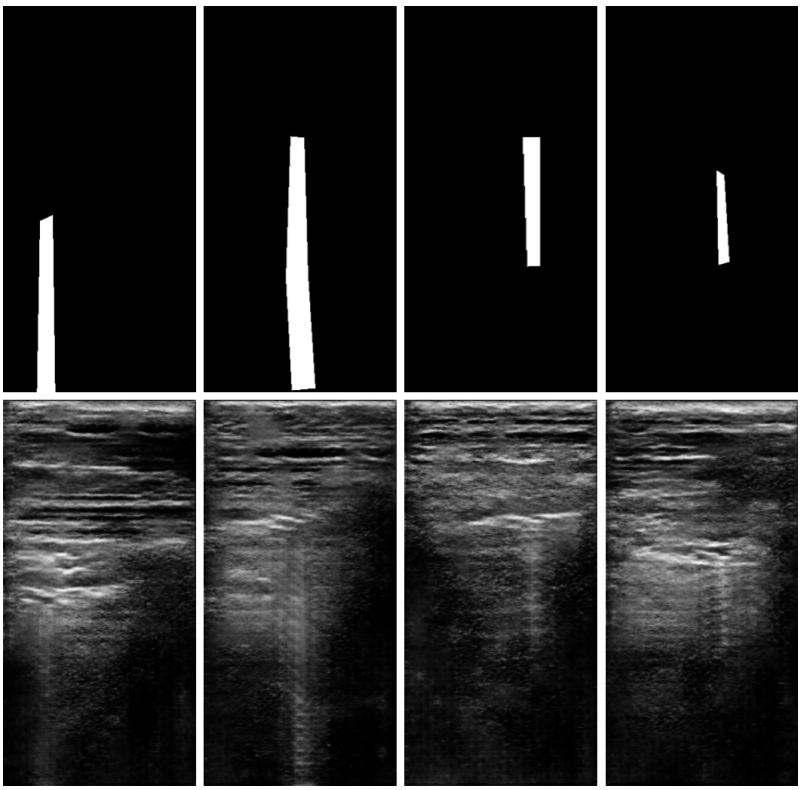

Figure 6 – 17 Pix2Pix: generated B-lines, first row is input, second row is generated image

6.5 Evaluation by classifier and experiments conclusion

As mentioned at the beginning of Section 6, the FID score only tells us how close the distribution of the generated images is to that of the real images. It doesn't tell us, however, if the generated images contain A-lines or B-lines artifacts. To evaluate if our models are capable of generating these artifacts with a certain level of quality, we employed a pre-trained classification model, one for each artifact. These models had been already pre-trained and tested on original images of A-lines or B-lines. However, due to the small size of the original training datasets, their performances were suboptimal. This is why we attempted to generate images with these artifacts in the first place. Our goal was to augment the original datasets and determine if the generated images could help improve the performance of the pre-trained classifiers. It's important to note that these models were not just some vanilla

versions, but were specifically optimized for A-lines and B-lines classification through spatial transformations and other augmentation techniques. They were optimized to achieve the best performance in classifying A-lines and B-lines, so augmenting the original datasets with generated images might not result in significant improvement. The models employed for classification utilized a ResNet-18 architecture [39].

We used each of our GAN to generate 500 images containing A-lines or B-lines, which were then used to augment the original datasets. During training for A-lines and B-lines classification, the negative class included images with the other type of line (A-lines for B-lines training and vice versa) and those without any visible lines (artifacts). The models were evaluated using 5-fold cross-validation. In this approach, the training dataset is split into 5 smaller sets. The model is then trained on some sets while being tested on others. This process is repeated with different sets, allowing for an estimation of the model's generalizability to unseen data and preventing overfitting [40]. The generated images were used exclusively in the training sets and were not used in the testing set. The testing set consisted solely of real images. To account for the variability across folds, values of accuracy, f1 score, precision, and recall were averaged.

Augmentation	Accuracy	F1 score	Precision	Recall
None	0.7629	0.7030	0.7134	0.7015
WGAN-GP-NM, Subsection 6.2	0.7594	0.6881	0.7255	0.6825
WGAN-GP-ED, Subsection 6.3.1	**0.7853**	0.6756	0.7093	0.6763
WGAN-GP-SC, Subsection 6.3.2	0.7733	**0.7234**	0.7416	0.7277
Pix2Pix, Subsection 6.4	0.7807	0.7230	**0.7432**	**0.7298**

Tab. 6 – 3 ResNet-18 performance when training with images of A-lines

Table 6 – 3 provides an overview of the ResNet-18 performance when trained on original images of A-lines compared to its performance when trained on an augmented dataset with generated images of A-lines. The results indicate that unconditional

synthesis with WGAN-GP did not improve the performance of the model. This is likely attributed to the lower quality of the generated images. As depicted in Figure 6 – 3, these images contain deformations, and A-lines artifacts are not clearly visible. Similarly, conditional synthesis with WGAN-GP, employing a generator with an encoder-decoder structure, did not yield improvements in the model performance. Although there was an enhancement in the quality of generated images, particularly in the generating of A-line artifacts, as can be seen in Figure 6 – 7, the overall results indicate no improvement in the model performance. On the other hand, augmenting a dataset with generated images from Pix2Pix and WGAN-GP models that utilize skip connections led to a slight improvement in ResNet-18 performance. This suggests that skip connections may play a role in guiding the generation of higher-quality images, particularly those containing A-line artifacts. It is also noteworthy that, thanks to these two approaches, the model achieved a better recall score, which measures the percentage of true positive cases the model correctly identifies. This is particularly important in the medical field, as misclassifying positive cases as negative can have serious consequences.

	Accuracy	F1 score	Precision	Recall
None	0.8206	0.6228	0.6389	0.6167
WGAN-GP-NM, Subsection 6.2	0.8479	0.6467	0.6336	0.6719
WGAN-GP-ED, Subsection 6.3.1	**0.8777**	**0.6851**	0.6909	**0.6872**
WGAN-GP-SC, Subsection 6.3.2	0.8719	0.6773	**0.7157**	0.6730
Pix2Pix, Subsection 6.4	0.8463	0.6090	0.6028	0.6367

Tab. 6 – 4 ResNet-18 performance when training with images of B-lines

Table 6 – 4 provides an overview of the ResNet-18 performance when trained on original images of B-lines compared to its performance when trained on an augmented dataset with generated images of B-lines. The results indicate that all our GANs contributed to some improvement in the model performance. Surprisingly, even unconditional WGAN-GP, despite the limitations observed in the representa-

tive images from Figure 6 – 5 (low quality and absence of B-line artifacts), managed to generate images that benefited the model. The most significant improvement was achieved with both types of conditional WGAN-GP approaches. These approaches were able to generate images with good quality and visible B-lines, as seen in Figures 6 – 9 and 6 – 13. These results indicate that using a binary mask as guidance during image generation played a significant role. Although Pix2Pix also utilizes a binary mask as input, it did not improve model performance to the same degree as the other approaches. This could be attributed to the presence of chessboard artifact, visible in the representative generated images in Figure 6 – 17. This artifact surrounds B-lines, making them blurry and hard to detect. Importantly, all approaches resulted in improved recall scores for the ResNet-18. As previously mentioned, high recall is particularly critical in medical image classification, as it minimizes the risk of missed positive cases.

	WGAN-GP-NM	WGAN-GP-ED	WGAN-GP-SC	Pix2Pix
FID-A	175	176	**152**	153
FID-B	167	127	**119**	155

Tab. 6 – 5 FID score summary

Table 6 – 5 summarizes the FID scores achieved by each of our models. The first row represents experiments generating images of A-lines, while the second row represents experiments generating images of B-lines. Upon examining these values, we observe a correlation with the results based on classifier performance improvement. For the generation of A-line images, both Pix2Pix and WGAN-GP with skip connections achieved the greatest improvement in classifier performance, accompanied by the lowest FID scores. Similarly, for the generation of B-line images, both conditional approaches with WGAN-GP demonstrated the highest improvement in classifier performance and obtained the lowest FID scores.

Based on these observations, we can conclude that Pix2Pix and WGAN-GP with

skip connections within the generator achieved the highest quality of generated A-line images. Conversely, for generated B-line images, the highest quality was achieved by both conditional approaches with WGAN-GP. While our results are promising and demonstrate the effectiveness of the chosen GAN architectures, it's crucial to acknowledge the limitations imposed by the relatively small training datasets utilized in this study. Generative Adversarial Networks, particularly those aiming for ultra-realistic image generation, are well-known to significantly benefit from large and diverse datasets, often containing thousands or even hundreds of thousands of images. Our dataset, consisting of only 1,074 images of A-lines and 346 images of B-lines, falls short of this requirement. Additionally, the origin of these images through video slicing introduces a bias, as some images are likely very similar to others. This redundancy can hinder the GAN's ability to learn the full range of variations present in real A-lines and B-lines, potentially limiting its ability to generalize effectively.

7 Conclusion

In this thesis, we aimed to address the critical issue of data scarcity in the lung ultrasound domain. Our research focused specifically on lung ultrasound images containing A-line or B-line artifacts. These artifacts play a vital role in lung health assessment, as A-lines typically indicate normal air-tissue interfaces, while B-lines are often associated with increased tissue density and potential lung pathologies.

Given the critical role of A-lines and B-lines in lung health assessment, deep learning methods hold great promise for their automated analysis. However, the effectiveness of deep learning hinges on a sufficiently large training dataset, which is not the case in the lung ultrasound domain. To overcome this challenge, traditional augmentation techniques like cropping, flipping, or rotations could have been used. However, such transformations could potentially alter the meaning of important structures within the image. For instance, applying a 90-degree rotation to an A-line might transform it into a B-line, leading to misinterpretation.

To address data scarcity, we employed various types of GANs. GANs are a class of deep learning models adept at creating novel data samples that share a high degree of similarity with the original dataset. We conducted two types of experiments to generate new lung ultrasound images. In the first, we used only random numbers from a gaussian distribution as input. In the second, we incorporated also a segmentation mask. In our experiments, we employed three different GANs, DCGAN, WGAN-GP, and Pix2Pix. To evaluate the quality of our generated images, we augmented the training dataset of a classifier with them. Then, we compared the performance of the classifier trained solely on real images with that of one trained on a dataset containing both real and generated images.

Despite a limited training dataset, our trained GANs improved classifier performance. A-line classification saw the most significant improvement (2% increase in accuracy, F1 score, precision, and recall) when augmenting with Pix2Pix-generated

images. Similarly, B-line classification benefited most from WGAN-GP with an encoder-decoder generator, achieving a notable 5% improvement across the same metrics. Based on these results, we have demonstrated that GANs are capable of generating images with sufficient quality to improve classifier performance.

Future work could address the challenge of generating ultrasound images by exploring various alternative GAN architectures or employing a pre-trained model as an additional discriminator. The normal discriminator would learn to distinguish between real and fake images in an adversarial manner, while the pre-trained classification or segmentation model would indicate to the generator whether the generated A-lines and B-lines artifacts are visible and of high quality. Additionally, other generative models like diffusion models or variational autoencoders could be employed to create new ultrasound images.

Bibliography

[1] Allan M Ross, Edward Genton, and Joseph H Holmes. Ultrasonic examination of the lung. *The Journal of laboratory and clinical medicine*, 72(4):556–564, 1968.

[2] Thomas J Marini, Deborah J Rubens, Yu T Zhao, Justin Weis, Timothy P O'connor, William H Novak, and Katherine A Kaproth-Joslin. Lung ultrasound: the essentials. *Radiology: Cardiothoracic Imaging*, 3(2):e200564, 2021.

[3] Arunangshu Chakraborty and Ashokka Balakrishnan. *A Practical Guide to Point of Care Ultrasound (POCUS)*. Springer, 2022.

[4] A Miller. Practical approach to lung ultrasound. *Bja Education*, 16(2):39–45, 2016.

[5] Ehsan Safai Zadeh, Christian Görg, Helmut Prosch, Daria Kifjak, Christoph Frank Dietrich, Christian B Laursen, and Hajo Findeisen. Lung ultrasound and pleural artifacts: A pictorial review. *Diagnostics*, 14(2):179, 2024.

[6] Daniel A Lichtenstein and Gilbert A Meziere. Relevance of lung ultrasound in the diagnosis of acute respiratory failure*: the blue protocol. *Chest*, 134(1):117–125, 2008.

[7] Bélaïd Bouhemad, Mao Zhang, Qin Lu, and Jean-Jacques Rouby. Clinical review: bedside lung ultrasound in critical care practice. *Critical care*, 11:1–9, 2007.

[8] Rohit Bhoil, Ajay Ahluwalia, Rajesh Chopra, Mukesh Surya, and Sabina Bhoil. Signs and lines in lung ultrasound. *Journal of Ultrasonography*, 21(86):225–233, 2021.

[9] Lars Ruthotto and Eldad Haber. An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44(2):e202100008, 2021.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[11] Tanujit Chakraborty, Ujjwal Reddy KS, Shraddha M Naik, Madhurima Panja, and Bayapureddy Manvitha. Ten years of generative adversarial nets (gans): a survey of the state-of-the-art. *Machine Learning: Science and Technology*, 5(1):011001, 2024.

[12] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[13] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE transactions on knowledge and data engineering*, 35(4):3313–3332, 2021.

[14] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[15] Sanjeet Singh Khanuja and Harmeet Kaur Khanuja. Gan challenges and optimal solutions. *International Research Journal of Engineering and Technology (IRJET)*, 8(10):836–840, 2021.

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

[17] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[18] Ali Borji. Pros and cons of gan evaluation measures: New developments. *Computer Vision and Image Understanding*, 215:103329, 2022.

[19] Yu Yu, Weibin Zhang, and Yun Deng. Frechet inception distance (fid) for eval-

uating gans. *China University of Mining Technology Beijing Graduate School*, 2021.

[20] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. *Neurocomputing*, 321:321–331, 2018.

[21] Alberto Montero, Elisenda Bonet-Carne, and Xavier Paolo Burgos-Artizzu. Generative adversarial networks to improve fetal brain fine-grained plane classification. *Sensors*, 21(23):7975, 2021.

[22] Tomoyuki Fujioka, Mio Mori, Kazunori Kubota, Yuka Kikuchi, Leona Katsuta, Mio Adachi, Goshi Oda, Tsuyoshi Nakagawa, Yoshio Kitazume, and Ukihide Tateishi. Breast ultrasound image synthesis using deep convolutional generative adversarial networks. *Diagnostics*, 9(4):176, 2019.

[23] Christoph Baur, Shadi Albarqouni, and Nassir Navab. Generating highly realistic images of skin lesions with gans. In *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis: First International Workshop, OR 2.0 2018, 5th International Workshop, CARE 2018, 7th International Workshop, CLIP 2018, Third International Workshop, ISIC 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16 and 20, 2018, Proceedings 5*, pages 260–267. Springer, 2018.

[24] Filippos Konidaris, Thanos Tagaris, Maria Sdraka, and Andreas Stafylopatis. Generative adversarial networks as an advanced data augmentation technique for mri data. In *VISIGRAPP (5: VISAPP)*, pages 48–59, 2019.

[25] Jiamin Liang, Xin Yang, Haoming Li, Yi Wang, Manh The Van, Haoran Dou, Chaoyu Chen, Jinghui Fang, Xiaowen Liang, Zixin Mai, et al. Synthesis and edition of ultrasound images via sketch guided progressive growing gans. In

2020 IEEE 17th international symposium on biomedical imaging (ISBI), pages 1793–1797. IEEE, 2020.

[26] He Zhao, Huiqi Li, Sebastian Maurer-Stroh, and Li Cheng. Synthesizing retinal and neuronal images with generative adversarial nets. *Medical image analysis*, 49:14–26, 2018.

[27] Kumar Abhishek and Ghassan Hamarneh. Mask2lesion: Mask-constrained adversarial skin lesion image synthesis. In *International workshop on simulation and synthesis in medical imaging*, pages 71–80. Springer, 2019.

[28] Yinhao Ren, Zhe Zhu, Yingzhou Li, Dehan Kong, Rui Hou, Lars J Grimm, Jeffery R Marks, and Joseph Y Lo. Mask embedding for realistic high-resolution medical image synthesis. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part VI 22*, pages 422–430. Springer, 2019.

[29] Maroš Hlibokỳ, Dmytro Lahunov, Samuel Gecík, and Marek Bundzel. Deep semantic segmentation models for lung ultrasound. In *2023 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, pages 98–103. IEEE, 2023.

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

[31] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[32] Bingqi Liu, Jiwei Lv, Xinyue Fan, Jie Luo, Tianyi Zou, et al. Application of an improved dcgan for image generation. *Mobile Information Systems*, 2022, 2022.

[33] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and

Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[34] Arthur Stéphanovitch, Ugo Tanielian, Benoît Cadre, Nicolas Klutchnikoff, and Gérard Biau. Optimal 1-wasserstein distance for wgans. *arXiv preprint arXiv:2201.02824*, 2022.

[35] Kalika Prasad and Ajit Iqbal Singh. Uniform continuity, lipschitz functions and their applications, 2014.

[36] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[38] Ugur Demir and Gozde Unal. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*, 2018.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[40] Mervyn Stone. Cross-validation: A review. *Statistics: A Journal of Theoretical and Applied Statistics*, 9(1):127–139, 1978.

Appendices

Appendix A User guide

Appendix B System guide

Appendix C CD Medium