Incorporating Shape Priors to Lung Segmentation with Generative Models

Ayoub EL HOUDRI

A report submitted as part of the requirements for the degree
of Master of Research & Master of Engineering
at Department of Computer Science
CY Cergy Paris Université
Cergy, France

October 2023

Supervisor Dr. Danny FRANCIS

Abstract

In the context of pathology screening through medical imaging, the integration of automated tools to expedite medical image analysis has become indispensable. Median Technologies is actively leveraging AI for medical imaging, with a specific focus on cancer screening. One of its standout offerings is the Lung Cancer Screening product, designed to facilitate the early identification of potential lung cancer cases through CT scans. This product aims to aid in the early identification of potential lung cancer cases through CT scans, with precise lung structure segmentation being key to its effectiveness.

Despite the seemingly straightforward nature of lung segmentation due to their distinctive appearance in CT scans, different confounding factors, such as gastric bubbles near the lungs and other anomalies, often lead to suboptimal segmentation outcomes.

This internship is dedicated exclusively to improving lung segmentation, considered as the foundational step in the cancer screening workflow. The primary objective is to achieve more accurate lung masks, particularly within challenging scenarios. To accomplish this, we have developed an architecture inspired by concepts of Wasserstein Generative Adversarial Networks (WGANs), a robust variant of GANs. By precisely customizing the losses, tuning segmentation process parameters, and incorporating prior knowledge into the segmentation model, we aim to enhance its accuracy.

The significance of this research subject lies in its potential to enhance the precision of lung cancer screening. A robust lung segmentation serves as the first step for subsequent analyses, enabling automated systems to effectively pinpoint potential cancerous lesions. Thus, this research establishes a strong foundation for improved cancer screening through medical imaging.

Acknowledgements

I want to extend my gratitude to my internship supervisor Dr. Danny FRANCIS for his invaluable guidance throughout my internship. Thanks to the Data Science & AI Team at Median Technologies for creating an excellent learning environment. I would also like to thank my professors whose teaching greatly assisted me during this internship. This experience has been pivotal for my studies, enhanced by the support, teamwork, and instruction I received.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

In today's healthcare landscape, medical imaging plays a pivotal role, assisting doctors in diagnosis and aiding researchers in studying diseases. With the emergence of Artificial Intelligence (AI) and data abundance, numerous companies and research centers have turned their focus toward developing cutting-edge techniques for medical imaging to support radiologists. However, this quest for precision has brought about its own challenges.

1.1 Medical Imaging Overview

Cancer screening, fundamentally, focuses on early detection for more effective treatment. Magnetic Resonance Imaging (MRI) technology, for instance, has undergone significant enhancements and is now capable of delivering highly detailed images used to identify small tumors or growths often invisible through conventional X-rays. Notably, MRI procedures are non-invasive and excel at capturing clear images of soft tissues, including detailed tumor tissues.

In the context of advancing the precise analysis of tumors and determination of their stage, Computerized Tomography (CT) scans have emerged as indispensable tools. By providing doctors with a high-resolution three-dimensional view of the body regions, CT scans facilitate the measurement of tumor size, location, and extent of spread. Such information is critical in tailoring the most appropriate treatment strategy, whether it entails surgery, radiation, or chemotherapy.

Medical imaging is considered a strong tool besides biomarker analysis in the field of cancer research. MRI and CT scans permit researchers to dive into the genetic and molecular alterations occurring within tumors. By integrating the information garnered from both medical imaging and biomarker analysis, we can gain a comprehensive macro and micro view of cancer. Thus contributing to a deeper understanding of cancer and the development of more precisely tailored treatments for individual patients.

More recently, the integration of AI into the field has triggered a revolution in medical imaging. Deep learning and machine learning algorithms have been meticulously crafted to swiftly and accurately analyze medical images. This technological innovation serves as a valuable aid to radiologists in terms of identifying anomalies or potential tumors within medical images and then expediting the diagnostic process.

1.2 Company Background

Median Technologies is a global company specializing in medical imaging and data science. They offer the iBiopsy platform, an AI-powered imaging solution for early cancer diagnosis and continuous patient monitoring. Additionally, Median provides iCRO solutions, harnessing proprietary AI and image analysis technologies for oncology clinical studies.

The company's headquarters is located in Sophia-Antipolis, Provence-Alpes-Côte d'Azur. With over 300 professionals, the company's primary focus is on Research and Development in the field of medical imaging for cancer screening. Median Technologies actively engages in various global cancer research activities, including publishing research articles in journals and conferences, as well as hosting seminars. Operating in both the American and Chinese markets, they are present in Boston, MA, and Shanghai, China.

There are two main activities, each assigned to a specific team, and the teams work on their respective activities, as detailed below:

The iBiopsy (Precision Diagnostic) team is working on developing a portfolio of innovative end-to-end AI and ML tech-based Software as Medical Devices (SaMDs) in indications where early diagnosis is needed. Their main goal is to enhance the quality of medical diagnostics through AI in Lung Cancer, Liver Cancer, and Liver Fibrosis Disease.

The first step of the process involves obtaining a CT scan of the patient. The CT scan is then converted into DICOM format and stored in the PACS which stands for Picture Archiving and Communication System, which is a medical imaging technology used primarily in healthcare organizations to securely store and transmit electronic images and clinical reports. The images within the PACS database undergo specific preprocessing to prepare them for use as input in the AI processing pipeline, whether it's for training or inference. A comprehensive DICOM report is generated and sent back to the PACS database, which can be accessed by radiologists to assist in their diagnosis (Figure 1.1).

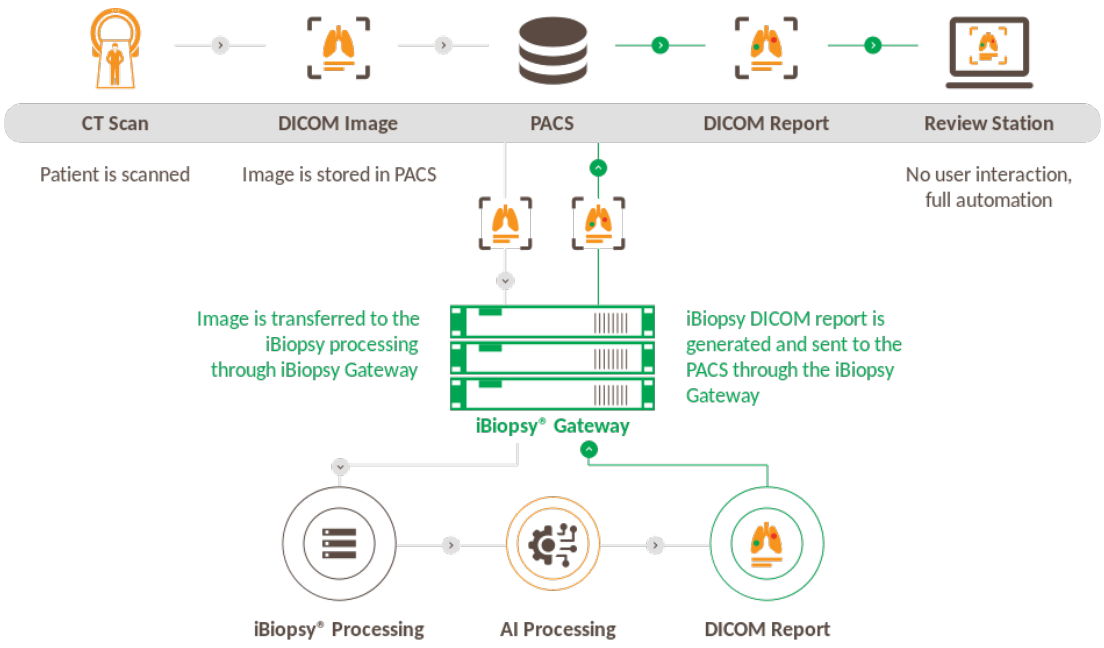

Figure 1.1: iBiopsy Workflow

The iCRO (Clinical Trials) team works on providing leading oncology clinical trial imaging

services, collaborating directly with different life sciences partners to accelerate the development, assessment, and delivery of cancer treatments.

Median iCRO assists clients throughout the clinical development process, beginning with protocol design and imaging criteria selection. They work to refine clinical strategies to improve product accuracy, cost efficiency, and asset value maximization. Their expertise in early-stage clinical studies includes providing access to advanced criteria and innovative imaging biomarkers for enhanced data. For Phase II-III trials, they offer efficient global clinical trial imaging solutions and handle regulatory submissions, along with supporting inspections by regulatory agencies.

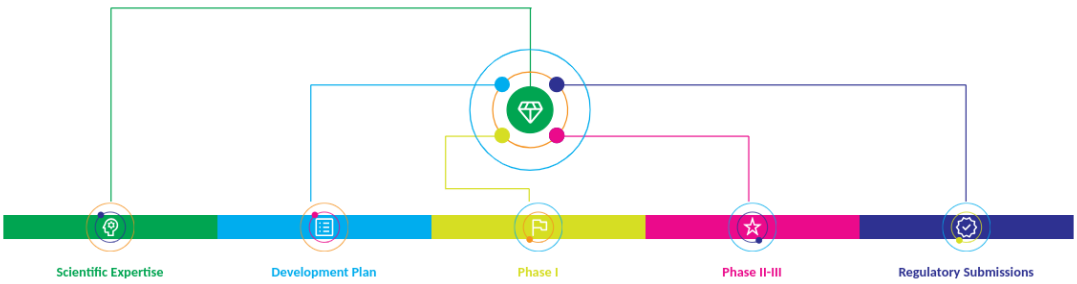

Figure 1.2: iCRO Workflow

Chapter 2

Image Segmentation

Medical imaging is essential in healthcare as it produces detailed visualizations of internal body structures through X-rays, MRI, and CT scans. Segmentation performance is evaluated using various metrics such as sensitivity, specificity, the Dice coefficient, Intersection over Union (IoU), and Hausdorff distance. The choice of the metric depends on the level of granularity and details needed to be captured in a given segmentation task, which is particularly important in the medical field.

2.1 Medical Image Segmentation

Image segmentation is a fundamental process in computer vision that involves classifying pixels (2D) or voxels (3D) with the goal of partitioning an image into distinct, meaningful regions. At its core, image segmentation consists of identifying and delineating objects or structures of interest within an image. In the field of computer vision, segmentation plays a vital role in tasks like object recognition, scene understanding, and image analysis.

In the context of medical image segmentation, highly capturing-details segmentation is very important, given that precise detection and localization of anatomical structures or pathological regions are essential for diagnosis. In 2D medical image segmentation, for example in MRI or CT scans, the objective is to identify and segment organs, tumors, or abnormalities within individual 2D slices extracted from the 3D scan.

When it comes to three-dimensional images, segmentation considers the volumetric nature of medical scans, such as 3D CT or MRI images. This continuity provides a more comprehensive understanding of the segmented object, and its surrounding regions, and serves as the foundation for more in-depth analysis.

Consider, for instance, the 3D MRI analysis of brain tumors. Accurate segmentation of tumor boundaries is essential. It allows radiologists to quantitatively assess tumor size, shape, and internal heterogeneity.

2.2 Scope and Objectives

In the same context, the internship I undertook within the R&D team at Median Technologies, located at their headquarters in Sophia-Antipolis. Dr. Danny Francis supervised this internship, with the main focus on 3D Lung Segmentation from CT Scans. This task serves as the initial

step in the cancer diagnosis pipeline, which is the AI Processing part in Figure 1.1, developed by the iBiopsy team.

The main challenge of the internship was linked to segmentation performance on certain patient data. To be specific, in some cases, patients had the gastric bubble positioned very close to the lung, causing the models to incorrectly label gastric bubble voxels as part of the lung. Consequently, this led to the presence of false positive voxels, potentially introducing complications in nodule detection and localization during the subsequent stages of the screening process. The final goal was to attain accurate segmentation, thus laying a robust first step for further analysis.

To address this issue, we explored various methods aimed at reducing the occurrence of false positives in our segmentation process. These methods consisted of getting benefits from shape priors through a feedback loop in adversarial training, the implementation of customized loss functions, and the use of an ensemble method.

Figure 2.1: Example of a CT slice from a patient with the gastric bubble positioned very close to the left lung

2.3 Evaluation of Segmentation Quality

Segmentation tasks involve the process of isolating single or multiple objects within an image. This is achieved by creating an appropriate segmentation mask, which is a 2D or 3D array matching the size and dimensions of the image containing the target object. To isolate the object of interest, we compute the dot-wise product between the mask and the original image. In the case of multiple objects, each individual object within the image containing multiple objects has its own mask, and the final result is obtained by combining these dot-wise products, effectively summing the contributions of each object's mask to the image.

Figure 2.2: A liver segmentation task, including a 2D slice from a 3D CT scan on the left, a binary liver segmentation mask in the middle, and the isolated liver resulting from the mask applied to the CT image on the right.

Assessing segmentation quality qualitatively by radiologists alone is insufficient, as it may overlook very small details, especially in 3D masks, potentially leading to precision issues. In such cases, various quantitative metrics are employed to ensure the accuracy of the final segmentation mask. This facilitates a more accurate comparison between the segmentation outcome mask and the ground truth mask.

Various metrics have been developed to evaluate segmentation by comparing the ground truth mask and the segmentation mask. Below, we present some of these metrics, with a particular focus on those used for our segmentation task.

2.3.1 DSC and IoU

The Dice Similarity Coefficient (DSC), also known as Sørensen-Dice [5], [29], is a widely used metric in image segmentation. It quantifies the overlap between two binary masks, typically the ground truth mask and the segmentation mask. The Dice score ranges from 0 to 1, with higher values indicating a greater degree of similarity between the two masks. It is calculated as follows:

$$DSC(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{2.1}$$

Intersection over Union (IoU), called also Jaccard coefficient [24], is another widely used metric in image segmentation, similar to the Dice score. It quantifies the overlap between two binary masks, the ground truth mask, and the segmentation mask. The IoU score measures the ratio of the intersection of the two masks to their union, providing a value between 0 and 1, with higher values indicating greater similarity. IoU is calculated as follows:

$$IoU(A, B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} = \frac{|A \cap B|}{|A \cup B|} \tag{2.2}$$

Where:

$|A \cap B|$ - Cardinality of intersection of mask A and B

$|A \cup B|$ - Cardinality of union of mask A and B

$|A|$ - Cardinality of mask A

$|B|$ - Cardinality in mask B

2.3.2 Hausdorff Distance (HD)

The Hausdorff distance [2] is a metric used in image analysis and computer vision to measure the dissimilarity or similarity between two sets of points or objects within an image.

It calculates the maximum distance from a point in one set to the nearest point in the other set and vice versa. In other words, it quantifies the maximum separation or closeness between two sets of points. More formally, for two sets A and B in a metric space:

$$d_{\mathrm{H}}(A, B) = \max \left\{ \sup_{a \in A} d(a, B), \sup_{b \in B} d(A, b) \right\} \tag{2.3}$$

Where:

$d(a, b)$ - Distance between point a in set A and point b in set B

$\max(d(a, B))$ - Maximum distance from a point in set A to the nearest point in set B

$\max(d(b, A))$ - Maximum distance from a point in set B to the nearest point in set A

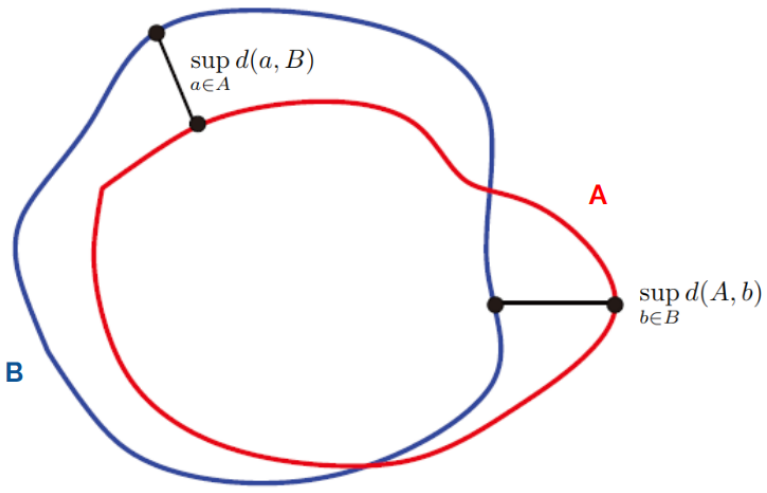

Figure 2.3: A visual description of Hausdorff distance between two masks A and B

This metric limitation is being less robust in handling errors caused by false positives and false negatives, unlike IoU and DSC. By prioritizing distance over region overlap, the Hausdorff distance takes into account more contour information, but can be more affected by misclassified pixels or regions.

Chapter 3

Data

Computed Tomography (CT) data provides the most detailed view of the internal structures of the human body among all medical imaging data to date. It uses X-rays to create cross-sectional images, commonly referred to as slices, of the body. These images are represented in a numerical format called Hounsfield Units (HU). HU values are used to attribute value to a voxel in standard 3D imaging, are a quantitative measure of tissue density and can help distinguish between different types of tissues. It ranges from -1000 HU for air, through 0 HU for water, to higher positive values for denser materials like bones, with a maximum of around 3000 HU. These values allow healthcare professionals to differentiate and analyze various anatomical structures, aiding in the diagnosis and treatment of a wide range of medical conditions.

Moreover, CT data is inherently three-dimensional in nature. Unlike traditional X-rays that provide 2D images, CT scanners capture a series of thin cross-sectional slices through the body, and these slices can be reconstructed into a 3D volume. This 3D aspect of CT data is very beneficial for visualizing a wide range of anatomical structures and helps precisely target areas of interest. Through CT scans, medical practitioners are provided with a comprehensive view of the patient's internal anatomy, which is essential for accurate and precise diagnosis.

3.1 Description

In the pursuit of precision, the imaging data used comprises CT scans, which are available within Median Technologies' Database for both training and testing. These images are available in

Tissue	**HU**
Bone	+1000
Liver	40 to 60
White matter	20 to 30
Grey matter	37 to 45
Blood	40
Muscle	10 to 40
Kidney	30
CSF	15
Water	0
Fat	-50 to -100
Air	-1000

Table 3.1: Body tissues and their corresponding HU values

Elements	Classes
Left lung	3
Right lung	4
Trachea	5
Background	0

Table 3.2: Segmentation mask components and their corresponding class for each voxel

various formats, including DICOM, npy, and zraw with mhd headers. The latter format is the one used for the experimental part of the internship.

We use datasets derived from subsets 0 and 1 of the LUNA16 database, which is openly available and was originally introduced in the article [27]. These datasets can be downloaded from the following link: https://luna16.grand-challenge.org/.

Our dataset contains CT scans from a total of 163 patients. For each CT scan in the dataset, a corresponding lung segmentation mask was created through an initial automatic segmentation process performed by our team. These masks underwent an in-house correction for a refined version.

The final version of our dataset is structured as pairs, each consisting of a CT image and its corresponding lung segmentation mask. 83 pairs are dedicated for training and validation purposes, while the remaining 80 pairs were used for testing and evaluation.

The CT images within our dataset shows variation along three axes, specifically identified as X, Y, and Z. Each dimension corresponds to a distinct view, as illustrated in Figure 3.1. These images have HU values for each voxel, as mentioned in the previous paragraph. These HU values span a range from -1,000 to 3,000 HU.

The segmentation masks within our dataset share a similar dimensionality, but they differ in terms of voxel label values. These label values are explained in Table 3.2.

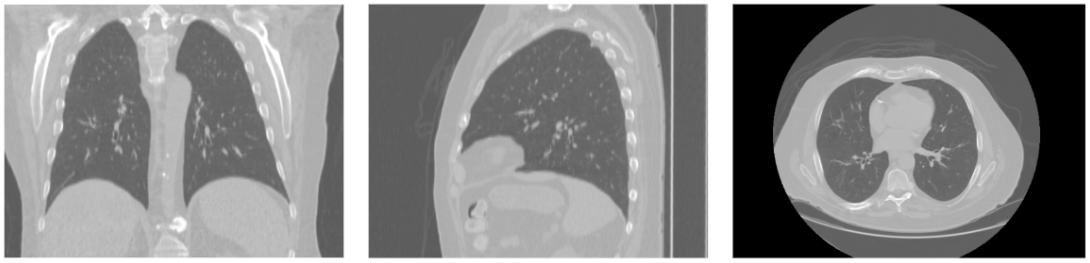

Figure 3.1: CT image views along the three axes (X, Y, and Z) from left to right

On the other side, there is the corresponding lung segmentation mask (see Figure 3.2), which matches the dimensions and size of the CT image. These masks have been annotated by radiologists and are divided into four classes, as described in Table 3.2.

Figure 3.2: Lung segmentation mask views along the three axes (X, Y, and Z) from left to right

3.2 Preprocessing

Preprocessing is necessary to standardize the data and extract the required information from the raw data presented above. The preprocessing steps are designed as a cascade of transformations to prepare the data for training, validation, and testing, which will be described in the following sections. The preprocessing pipeline is presented in Figure 3.3.

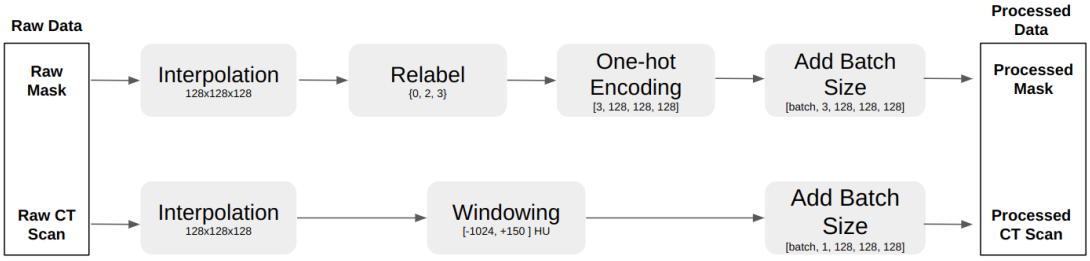

Figure 3.3: Preprocessing pipeline for the CT images and their corresponding masks

Trilinear Interpolation is the first step, where the ground truth masks and CT images are resized from their initial size to 128×128×128 using trilinear interpolation following the 3 axes X, Y and Z.

Relabeling involves assigning new labels to the initial labels presented in Table 3.2. The output is a mask where background voxels retain label 0, left lung voxels are assigned to the new class 2, right lung voxels to the new class 3.

Windowing involves sliding a window over the CT image to restrict all HU values to fall within a specified range, which is the window width. In this case, the range is set to [-1024, +150] HU. This process effectively narrows down the range of information that is retained.

One-hot Encoding is a transformation that increases the segmentation mask dimensionality to three masks, each with a size of 128×128×128, matching the initial size. These three masks represent the left lung, right lung, and background, where voxels belonging to the respective structures are denoted as 1, and 0 otherwise. This transformation applies only to segmentation masks, CT images are kept unchanged.

Adding Batch represents the final transformation in the process. It applies to both the masks and the CT images, introducing a new dimension corresponding to the batch size. The specific batch size is used later in the experimental phase.

Chapter 4

Segmentation Methods

Classical image segmentation methods have played a significant role in computer vision for decades. They consist of using a wide range of techniques that aim to partition an image into meaningful regions or objects. However, these methods, while effective in many scenarios, are not without their limitations. These shortcomings have spurred the adoption of deep learning techniques, which have revolutionized the field of image segmentation and led to better precision and adaptation.

In this section, we will highlight the classical image segmentation methods, highlighting their drawbacks, and explore how these challenges have driven the shift towards leveraging deep learning for more robust and accurate segmentation tasks.

4.1 Classical Methods

Threshold methods are among the basic image segmentation techniques. They operate by defining a threshold value and then categorizing each pixel in an image as either foreground or background based on their intensity values compared to a threshold. The first threshold [17], These static methods often struggle with noisy images, as they may produce incorrect results due to fluctuations in pixel values. Additionally, selecting an appropriate threshold value can be challenging, as it frequently depends on the specific characteristics of the image, such as lighting conditions and object properties.

Region-based segmentation methods focus on grouping pixels that share similar attributes, such as color, texture, or intensity, into distinct regions or segments. This approach aims to identify coherent regions within an image. However, region-based methods are sensitive to the initial selection of seed points, which can significantly impact the segmentation quality. Over-segmentation, where an image is divided into too many small regions, and under-segmentation, where regions are overly large, are common issues. Fine-tuning parameters is often necessary to achieve nearly optimal results. In [20], the authors give an overview of different region-based segmentation methods and provide a comparison between them.

Edge-based segmentation techniques concentrate on identifying edges or boundaries in an image. They rely on edge detection algorithms [35] to locate fast changes in pixel intensity, which often correspond to object boundaries. While these methods are effective at detecting edges, they may produce false edges in noisy images and can suffer from incomplete or disconnected contours. Moreover, the edges detected may not necessarily correspond to meaningful object boundaries, making them less suitable for tasks requiring precise object segmentation.

Watershed-based segmentation [4] deals with the image as a topographic landscape and simulates the process of flooding to delineate object boundaries. Although it can be effective for some tasks, watershed segmentation often faces challenges such as over-segmentation, especially in textured regions, where fine details can lead to numerous small segments. A precise preprocessing is typically required to avoid excessive segmentation, and the method can be sensitive to noise in the image.

Clustering methods group pixels into clusters based on feature similarity such as color, texture, or intensity. While they provide the possibility of defining the number of clusters, selecting an appropriate number of clusters can be a complex task. Moreover, the accuracy of clustering methods is tied to the choice of similarity metrics, which can influence their performance.

These methods can provide reasonably good segmentation masks, but they exhibit different weaknesses in terms of precision, which is a requirement in medical image segmentation, as explained in a previous section.

4.2 Deep Learning Methods

4.2.1 Standard Convolutional Neural Networks

The emergence of deep learning, particularly Convolutional Neural Networks (CNNs) and the backpropagation algorithm [21] have revolutionized image segmentation, offering superior results compared to traditional methods. CNNs excel at capturing complex image patterns and object boundaries by automatically learning hierarchical features from data [19] through a sequence of convolutions and pooling (see Figure 4.1)

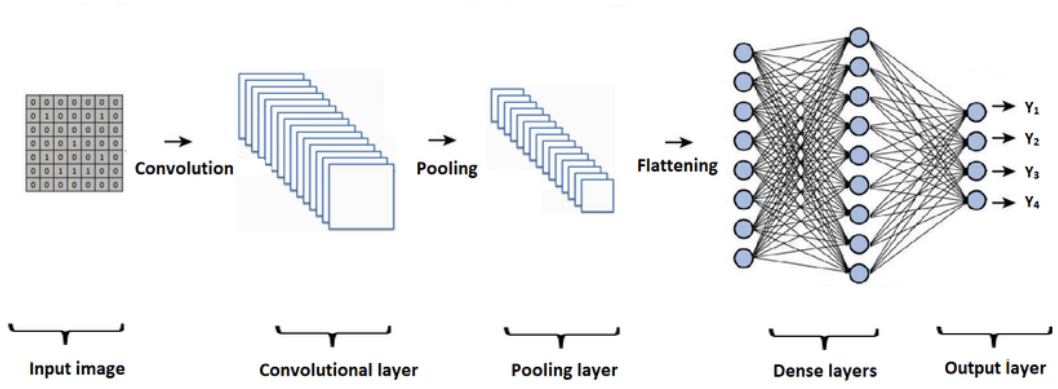

Figure 4.1: Layers comprising a Convolutional Neural Network

These advancements have proven especially valuable in precision-demanding applications like medical imaging. In tasks such as tumor detection in MRI scans [11] or organ segmentation in CT images [26], deep learning consistently outperformed traditional methods.

4.2.2 Convolutional Autoencoders

Different models were adapted from general segmentation tasks to be used for medical image segmentation. They are primarily based on Convolutional Autoencoders, which are essentially standard CNNs with an autoencoder architecture [18]. This architecture leverages the encoding of learned features into a latent space, which can later be decoded. This approach benefits from inferring a standard primary shape between the data within the same distribution.

Models like U-Net [26], U-Net++ [34], V-Net [23], and SegResNet [25] originally emerged for 2D medical image segmentation tasks and sometimes for 3D images. However, they also served as inspiration for the development of new models only for 3D medical image segmentation, such as 3D U-Net [3], among others. The architectural process remains the same, with the only difference being that 2D convolution blocks were changed to 3D convolution blocks to handle volumetric data.

What these models have in common is the use of skip connections in their architecture. These skip connections are designed to prevent information loss in the encoder section, facilitating the creation of tailored segmentation masks for each CT image. As the network processes the data, high-level information is incrementally captured in the initial layers and subsequently combined with their symmetrical layers in the decoder (see Figure 4.2). This technique effectively prevents information degradation during the compression into latent space. Thanks to these skip connections, these models achieve exceptionally precise segmentation by generating highly accurate segmentation masks, compared to other models.

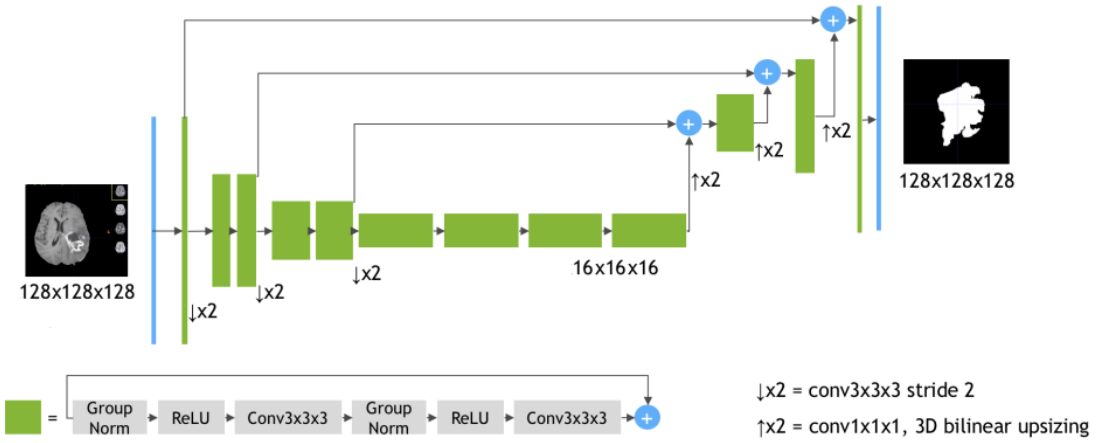

Figure 4.2: SegResNet [25] architecture used in a task of brain segmentation from 3D CT images

SegResNet was initially used for a 3D MRI brain tumor segmentation task [25], and it demonstrated competitive performance in various challenges when compared to 3D U-Net and other segmentation models. During this internship, we utilized SegResNet as the foundational model for the task of lung segmentation from 3D CT images. We applied different loss functions to train and evaluate the model using data from 163 patients.

Chapter 5

Implementations and Experiments

In this section, we will explore various implementations initially employed for the segmentation task. Initially, a foundational model was used to accomplish our segmentation task. Subsequently, in pursuit of enhancing segmentation accuracy, we integrated this model into different adversarial training setups, employing it as a generator in conjunction with five distinct discriminator architectures. To refine the segmentation masks and provide more intricate details, an optimization of the latent space was implemented. Finally, an ensemble method was employed to produce a final version of the segmentation mask from different predictions.

5.1 Our Base Segmentation Model

We utilized SegResNet, the architecture of which is presented in Figure 4.2. The model underwent training using the preprocessed dataset detailed in Chapter 3. The underlying principle of this model revolves around encoding information within a latent space through the encoder component and subsequently retrieving this information from the latent space via the decoder component. This information is decoded and post-processed to have a binary mask.

Architecture

The encoder architecture is made from a series of individual blocks, called ResNet blocks [12], that helps in avoiding gradient vanishing issues through the residual skip connections, each composed of three sequential operations: Group Normalization [33] followed by ReLU activation [8], and a 3×3×3 kernel convolution applied to the input with a stride of 2. This structure ensures a systematic processing of the input data. The organization of these blocks follows a hierarchical pattern.

Initially, the input image, with dimensions of 128×128×128, is processed by the first layer, which contains just one block. Subsequently, the output from the first layer is fed into the second layer, which is made of two of these blocks. Following this, the output continues into the third layer, which also includes two blocks. Finally, the output is directed to the fourth and final layer, where four blocks are used. This encoding process produces a set of feature maps with dimensions identical to those of the original CT image, but with a reduced size of 16×16×16, and corresponds to our latent space.

The decoder, in turn, consists of the same blocks. However, instead of a 3×3×3 convolution,

we use a 1×1×1 convolution followed by 3D bilinear upsizing to increase the size of the feature maps retrieved from the latent space. The decoder's architecture is made of three layers with one block each: the first layer takes input from the latent space, and its output is then passed to the second block, which in turn feeds the third block. The sizes increase symmetrically in the other direction compared to the first three blocks in the encoder.

The specificity of this U-Net-like architecture is its use of skip connections, used to prevent the loss of high-level features collected from the first three layers of the decoder. These features are progressively concatenated with the feature maps from the latent space at the final layer of the encoder, as well as inputs to the three layers of the decoder, as highlighted in Figure 4.2.

As a result, the decoder generates a segmentation mask with the same dimensions as the CT image (128×128×128). This output contains three channels, same as the dimensionality of approximated ground truth mask for the left lung, right lung, and background. Each value assigned to a voxel within a channel represents the probability of that voxel belonging to the corresponding segmented element.

In this context, the vector associated to each output voxel serve as a probability vector, with each channel approximating the ground truth values of 0 for background, 1 for the left lung, and 2 for the right lung.

Loss Function

The optimized loss function used for this model in a normal training setup is Cross-Entropy (CE), which can be calculated as:

$$\mathbb{CE}(A, B) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\left(B_{ij}\log(A_{ij}) + (1 - B_{ij})\log(1 - A_{ij})\right) \tag{5.1}$$

Where:

- $\mathbb{CE}(A, B)$ - Cross-entropy between generated mask A and ground-truth mask B
- N - Number of voxels in the masks A and B
- C - Number of classes or channels in the masks A and B (C is set to 3 in our case)
- A_{ij} - Voxel value in the generated mask A for voxel i and channel j
- B_{ij} - Voxel value in the ground-truth mask B for voxel i and channel j

$\mathbb{CE}$ loss, introduced by Shannon in the context of telecommunication [28], aims to quantify the disparities in information content between the real and predicted values. It is widely employed in classification tasks, including segmentation, as a loss function to minimize. It is differentiable, measures dissimilarity among classes, and can easily handle multi-class problems.

5.2 Our Base Model in Adversarial Training Setup

With the goal of improving segmentation, we worked on using the model (SegResNet) as a generator in a generative adversarial neural network (GANs) [10] setup to benefit from the feedback offered by the discriminator through adversarial training.

GANs consist of putting two neural networks in competition (a zero-sum game) [10], where

the advancement of one network comes at the expense of the other. Through training on a dataset, this approach acquires the capability to produce new data with the same statistical characteristics as the original training data.

The two networks are called generator, and discriminator, where the generator works to improve its ability to produce realistic data to fool the discriminator, while the discriminator continually refines its ability to differentiate real from fake data. This adversarial process drives both networks to improve their performance, often resulting in the generation of high-quality data samples that are increasingly difficult to distinguish from real ones.

GANs may suffer from mode collapse, where the generator fails to produce diverse outputs, instead generating a limited set of similar samples. This issue can hamper the ability of GANs to capture the full complexity of the data distribution.

In our task, we employ a GAN in a distinct context, separate from data reconstruction or augmentation. Instead, we utilize it to generate segmentation masks using the SegResNet generator, as elaborated in Section 5.1, for all the experiments outlined in this report. These generated masks undergo evaluation by a discriminator to measure their proximity to the desired outcome. The discriminator's loss function provides feedback to the generator, establishing a competitive dynamic between both networks.

We use a loss function inspired by the Wasserstein distance, as introduced here [32] and also used as a solution to mode collapse in Wasserstein GAN (WGAN) [1]. The Wasserstein distance formula is given in (5.2).

$$W(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \mathbb{E}_{(x,y)\sim\gamma} d(x, y) \right) \tag{5.2}$$

Where:

μ, ν - Two distributions

$\Gamma(\mu, \nu)$ - Set of all couplings of distribution μ and ν

γ - A coupling distribution in $\Gamma(\mu, \nu)$

$d(x, y)$ - Distance between x and y

Our objective is to train the generator to achieve a high level of accuracy by using discriminator feedback. The discriminator's role is to assign high scores to a distribution of inputs involving ground truth masks and low scores to generated masks, justifying the use of Wasserstein distance-inspired loss function. This process makes the discriminator stronger and enables the generator to produce refined segmentation masks.

These experiments were inspired by the work presented in [31], where the architectures were originally applied to 2D lung segmentation from CT scans. In our work, we adapted these architectures by using 3D convolutions to perform direct 3D lung segmentation on 3D CT images.

Our generator remains the same throughout all experiments, while we explore five different discriminator architectures in our quest to enhance accuracy and reduce the gastric bubble in the segmentation mask.

In all discriminator architectures, we employ LeakyReLU activation [22], as defined in (5.3), with a slope parameter of $\alpha = 0.1$.

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ \alpha x & \text{otherwise} \end{cases} \tag{5.3}$$

We also use batch normalization, as introduced in [14], to accelerate and stabilize the training process.

The mean of a batch B with a size of m is computed according to the formula (5.4), while the variance is calculated as shown in formula (5.5). Batch normalization is then applied to the elements x_i within a batch B using the formula (5.6).

Additionally, we include the parameter ϵ in the denominator for numerical stability, which is useful if $\sigma_B^{(k)}$ becomes very small.

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{5.4}$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{5.5}$$

$$\hat{x}_i^{(k)} = \frac{x_i^{(k)} - \mu_B^{(k)}}{\sqrt{\left(\sigma_B^{(k)}\right)^2 + \epsilon}} \tag{5.6}$$

Dropout [30], is a regularization technique that enforces sparse activation for the concerned layer. This encourages the network to acquire a sparse representation of the learned features, then preventing overfitting. In all our experiments, we apply dropout with a dropout rate of $P_{dropout} = 0.25$. This means that 25% of the neurones activity of the model are set to zero, as described in (5.7).

$$\hat{a}_{ij} = \begin{cases} 0 & \text{with probability } P_{dropout} \\ a_{ij} & \text{with probability } 1 - P_{dropout} \end{cases} \tag{5.7}$$

All of our models are trained from scratch, whether the generator is used alone in the model or as part of a GAN setup. We do not use any pre-trained weights in our training process.

Our weights are initialized following Xavier initialization [9] which consists of initializing the weights such that the variance of the activations are the same across every layer, which prevents gradient vanishing or gradient exploding.

5.2.1 Basic Discriminator

In this subsection, we will implement the basic idea of employing SegResNet as a generator alongside a naive discriminator. We will refer to the entire model, highlighted in Figure 5.1,

consisting of the generator (G) and the basic discriminator (D), as GAN_B.

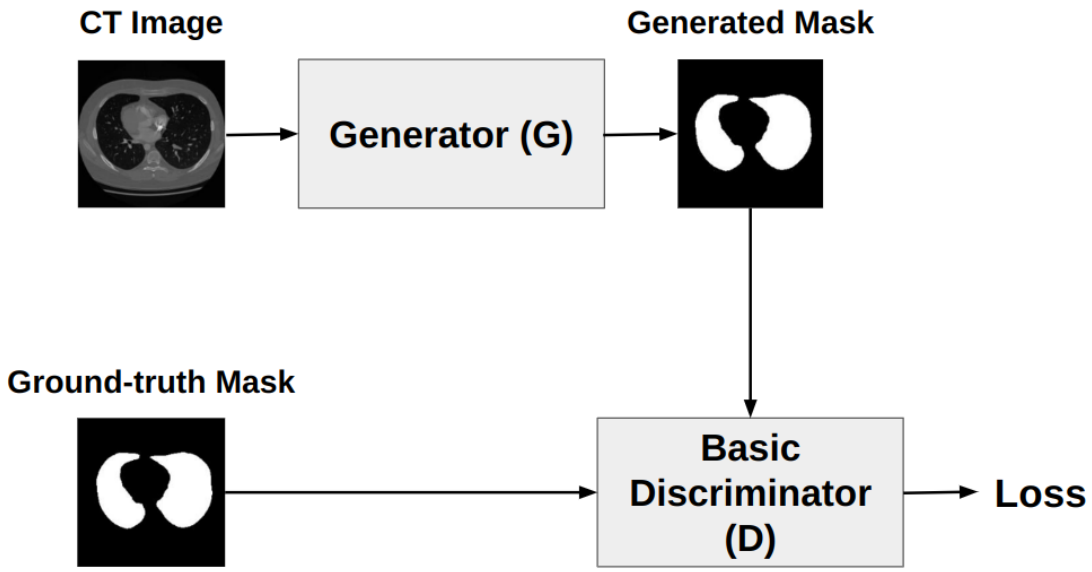

Figure 5.1: GAN_B architecture with a generator (G) and basic discriminator (D)

Architecture

The discriminator takes as input the segmentation masks generated by the generator for a batch, as well as the batch of ground truth masks corresponding to the CT images used as input for the generator. Both the generated masks and ground truth masks are processed by the discriminator, resulting in a set of batch scores for each. These scores are used to compute the loss function, which is explained in the subsection below.

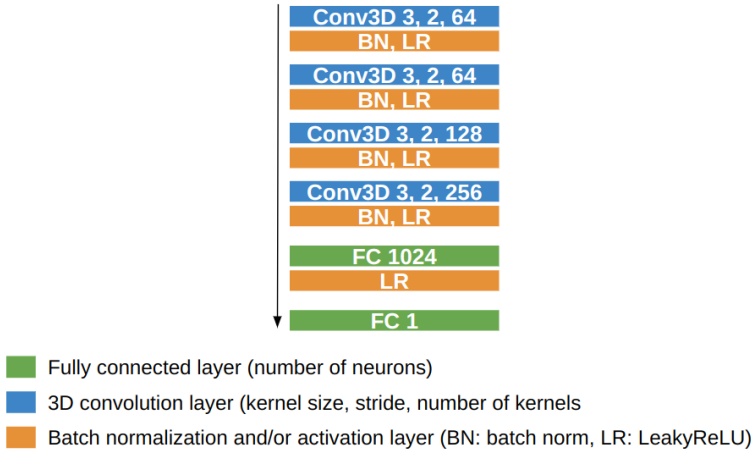

Figure 5.2: Basic discriminator architecture

Loss Function

The loss function employed for the generator comprises two components: CE term (5.1), which enforces segmentation, and an additional component that considers the mean score provided by the discriminator for the generated mask inputs. This latter component is responsible for the adversarial competition, with a goal of having a basic segmentation and then bringing the generated masks closer to the ground truth masks. It is given in (5.8).

The loss function used for the discriminator is based on the distance between the means of two distributions: the distribution of the ground truth masks and the distribution of the generated masks within a batch. This loss is directly inspired by the Wasserstein distance, as detailed above in Section 5.2. The objective is to augment this difference, or inversely, decrease the negative value of this difference. By doing so, the aim is to train the discriminator to assign high scores to ground truth values and lower scores to generated values, thereby enhancing the discriminator's ability to distinguish between ground truth masks and generated masks. Its mathematical expression is provided in (5.9).

$$\mathcal{L}_G = \mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x \sim P_z}[D(G(x))] \tag{5.8}$$

$$\mathcal{L}_D = \mathbb{E}_{x \sim P_z}[D(G(x))] - \mathbb{E}_{y \sim P_{\text{Mask}_{\text{GT}}}}[D(y)] \tag{5.9}$$

Where:

Mask_{GT} - A batch of ground-truth masks

$G(\bullet)$ - Generator function to generate masks from CT images x using the generator G

$D(\bullet)$ - Discriminator function to score masks ($x \sim P_z$)

$\mathbb{E}(\bullet)$ - Mean value of a distribution

P_z - Distribution of CT images in a batch

$P_{Mask_{GT}}$ - Distribution of ground truth masks in a batch

Both loss functions are minimized during training, with the discriminator being the starting point. Initially, there is the minimization of the term $\mathbb{E}_{x \sim P_z}[D(G(x))]$ and the maximization of $\mathbb{E}_{y \sim P_{\text{Mask}_{GT}}}[D(y)]$ within the discriminator loss, as explained above. On the other hand, the generator's loss is minimized subsequently, leading to a decrease of cross-entropy for a forced segmentation, while $\mathbb{E}_{x \sim P_z}[D(G(x))]$ is maximized. This indicates that the generator learns to fool the discriminator by assigning high scores to the generated masks, in contrast to the discriminator's objective. This competition results in a refined generated mask at the end of training.

5.2.2 Product Discriminator

To enhance the performance achieved by our segmentation model, our implementation in this section incorporates the evaluation of voxel values within the lung region by the discriminator, as opposed to using only binary values as input as it is the case for GAN_B. Therefore, we employ SegResNet as a generator and introduce a product discriminator. The complete model, as depicted in Figure 5.3, encompasses both the generator (G) and the product discriminator (D), and we will refer to it as GAN_P.

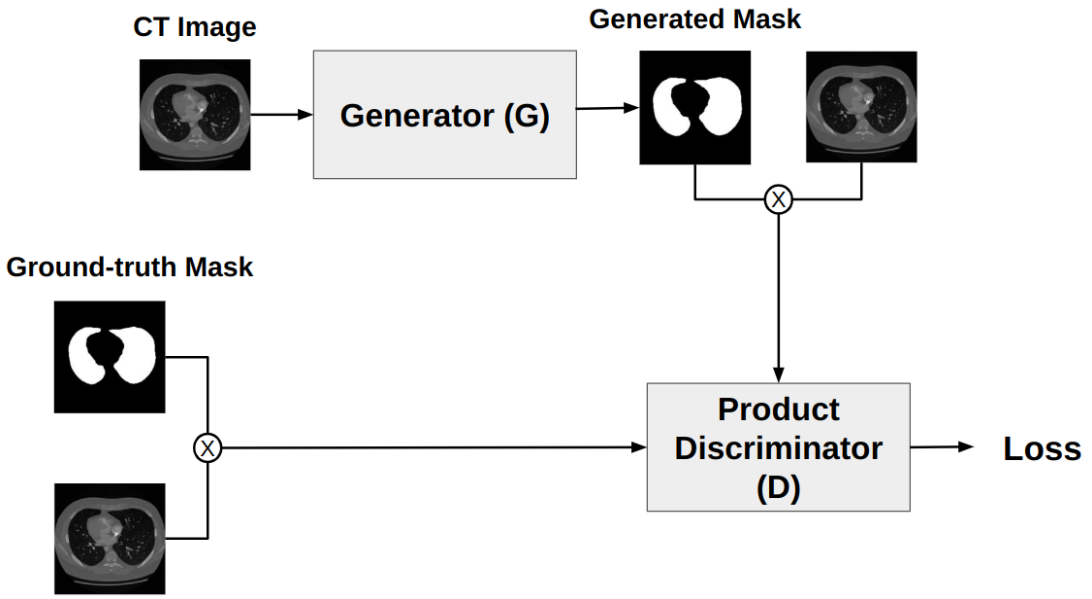

Figure 5.3: GAN_P architecture with a generator (G) and product discriminator (D)

Architecture

The architecture of the product discriminator is identical to that of the basic discriminator (see Figure 5.2). The key difference lies in the consideration of the element-wise product between the generated mask from the generator and the CT image, resulting in an image containing only lung voxels while blacking out voxels outside the lung region. This operation is also done between the ground truth mask and its corresponding CT image. Subsequently, both the real segmented lung and the segmented lung using the generated mask are input to the discriminator. The discriminator performs the same operations as mentioned for the basic discriminator.

Loss Function

The same loss functions employed for the basic discriminator are also used for the product discriminator. The only variation lies in the inputs, as previously mentioned. As a result, the generator's loss is written in (5.10), and the discriminator's loss is expressed in (5.11).

$$\mathcal{L}_G = \mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x\sim P_z}[D(G(x) \odot x)] \tag{5.10}$$

$$\mathcal{L}_D = \mathbb{E}_{x\sim P_z}[D(G(x) \odot x)] - \mathbb{E}_{y\sim P_{\text{Mask}_{\text{GT}}}}[D(y \odot x)] \tag{5.11}$$

Where:

$\odot$ - Element-wise product operator, called also Hadamard product operator

Mask_{GT} - A batch of ground-truth masks

$G(\bullet)$ - Generator function to generate masks from CT images x using the generator G

$D(\bullet)$ - Discriminator function to score the element-wise product of masks and their CT images

$\mathbb{E}(\bullet)$ - Mean value of a distribution

P_z - Distribution of CT images in a batch

$P_{Mask_{GT}}$ - Distribution of ground truth masks in a batch

5.2.3 Early Fusion Discriminator

In the context of the early fusion discriminator, our goal is to implement a novel architecture to improve our base segmentation model. Instead of relying on a discriminator that employs binary masks or the segmented lung, we utilize inputs that consist of the complete CT image concatenated with the generated mask. This input is then compared to the concatenation of the ground truth mask and its corresponding CT image through the discriminator, as shown in Figure 5.4. This early stage concatenation is the basis for referring to the discriminator as an early fusion discriminator. The whole model made of the generator and the discriminator will be identified as GAN_{EF}.

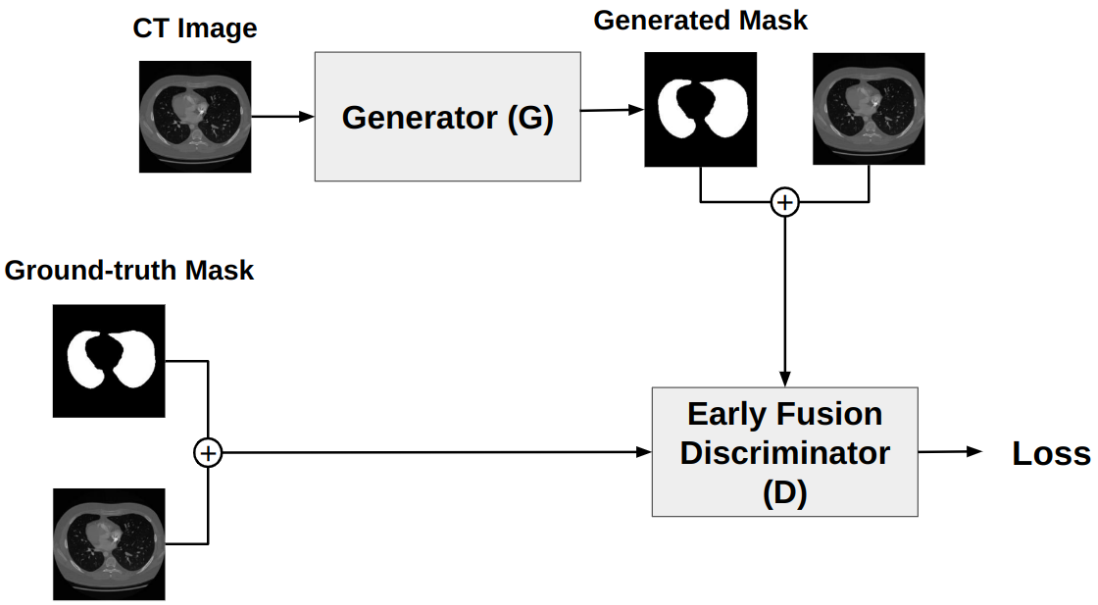

Figure 5.4: GAN_{EF} architecture with a generator (G) and early fusion discriminator (D)

Architecture

The architecture of the early fusion discriminator maintains an identical structure to that of the basic discriminator, as depicted in Figure 5.2. The key difference lies in the use of concatenation between the mask generated by the generator and the CT image. This concatenation yields an image that takes into account both the generated mask and the CT image. This design choice is driven by the recognition that incorporating the CT image alongside its associated mask provides valuable contextual information about the lung's surrounding area. This additional information can prove beneficial to the discriminator by aiding in the assessment of the dissimilarity between the ground truth masks and the generated masks. A similar concatenation operation is carried out between the ground truth mask and its corresponding CT image.

Loss Function

The loss functions for both the generator and the discriminator are very similar to those of the basic discriminator and the product discriminator. However, since the discriminator's inputs differ from those of the discriminators detailed above, the loss functions experience a change in inputs, as outlined below. In these equations, (5.12) represents the generator loss, and (5.13) corresponds to the early fusion discriminator loss.

$$\mathcal{L}_G = \mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x \sim P_z}[D(G(x) \oplus x)] \tag{5.12}$$

$$\mathcal{L}_D = \mathbb{E}_{x \sim P_z}[D(G(x) \oplus x)] - \mathbb{E}_{y \sim P_{\text{Mask}_{\text{GT}}}}[D(y \oplus x)] \tag{5.13}$$

Where:

$\oplus$ - Concatenation operator following the channels dimension

Mask_{GT} - A batch of ground-truth masks

$G(\bullet)$ - Generator function to generate masks from CT images

$D(\bullet)$ - Discriminator function to score the concatenation of masks and their corresponding CT images

$\mathbb{E}(\bullet)$ - Mean value of a distribution

P_z - Distribution of CT images in a batch

$P_{Mask_{GT}}$ - Distribution of ground truth masks in a batch

5.2.4 Late Fusion Discriminator

The late fusion discriminator has a distinct architecture compared to the other discriminators presented so far. It consists of two branches: one branch takes CT images as input, while the other focuses on the masks. The features extracted by these two branches are concatenated into feature maps, which undergo further processing within the discriminator (see Figure 5.5), that learns to score its inputs. We will refer to this model as GAN_{LF}.

The underlying concept behind this implementation is to extract significant features from both the CT Image and its corresponding mask, enabling us to obtain enhanced information from both sources. These features are concatenated and processed through additional layers within the discriminator. This approach stands in contrast to early fusion, which concatenates the CT image and its mask from the outset. Late fusion, by contrast, can capture more detailed information and leverage these important features effectively.

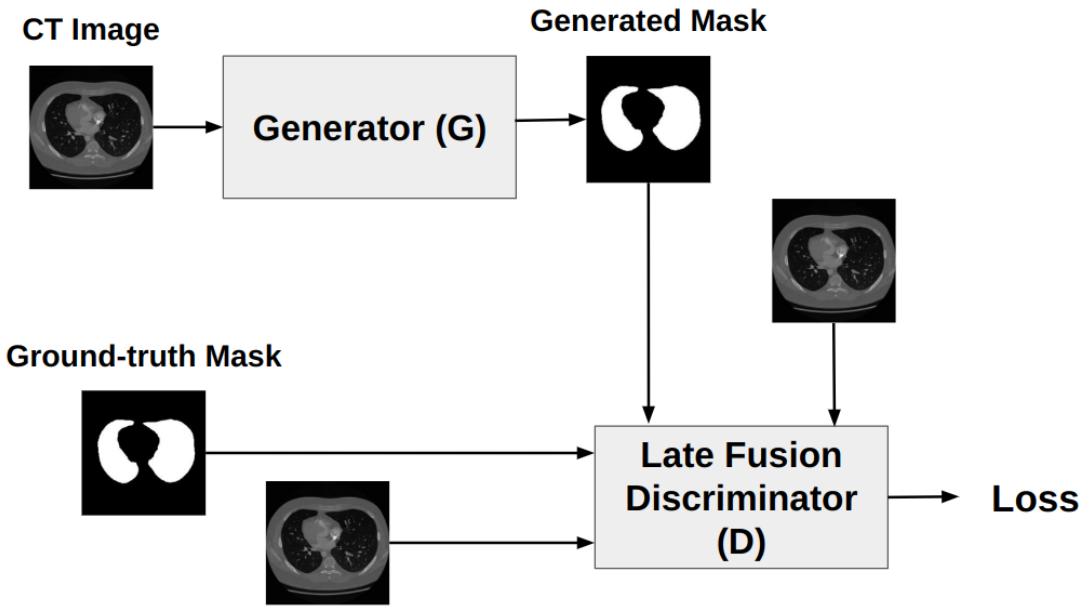

Figure 5.5: GAN_{LF} architecture with a generator (G) and late fusion discriminator (D)

Architecture

The architecture of the late fusion discriminator is detailed in this subsection and presented in Figure 5.6. The left branch, made of one convolutional layer, takes the segmentation mask as input, while the right branch, consisting of two convolutional layers, takes the CT image as input. After passing through these layers, the feature maps from both branches are concatenated. This concatenation is performed along the channels dimension, resulting in feature maps with an increased number of channels.

The concatenated feature maps are then fed to a final layer composed of a series of convolutional operations, ending with two fully connected layers. The output of this discriminator is a score used to discriminate between pairs: the generated mask and its corresponding CT image, and the pair consisting of the ground-truth mask and its corresponding CT image.

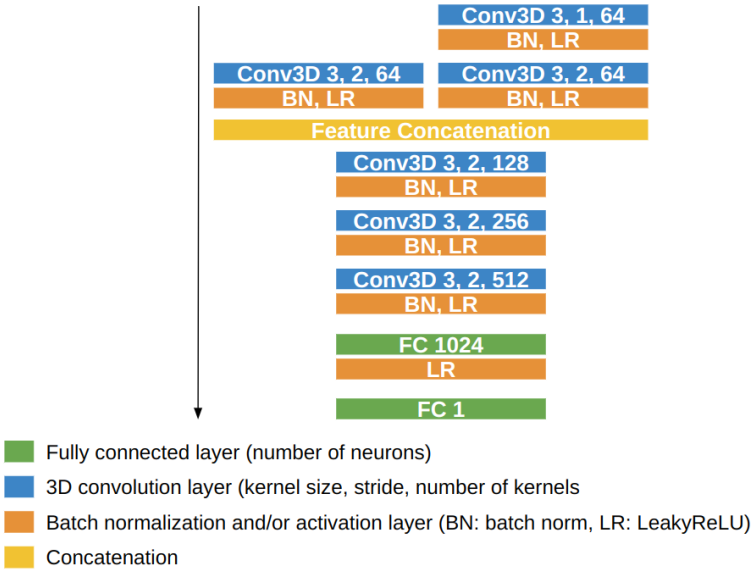

Figure 5.6: Late fusion discriminator architecture

Loss Function

For both the generator and the discriminator, the losses are given as (5.14) and (5.15), respectively. These losses are used in the same context as the loss functions used for the basic discriminator and the previous losses. However, the slight difference is in the discriminator's inputs.

Instead of a single input for one branch of the discriminator, we now use two inputs for each of the two branches of the discriminator. These input pairs are denoted as $(G(x), x)$ for the generated mask and the CT image and (y, x) for the ground truth mask and the CT image, where x is from the distribution P_z of CT images, and y is from the distribution $P_{\text{Mask}_{\text{GT}}}$ of ground truth masks.

$$\mathcal{L}_G = \mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x \sim P_z}[D(G(x), x)] \tag{5.14}$$

$$\mathcal{L}_D = \mathbb{E}_{x \sim P_z}[D(G(x), x)] - \mathbb{E}_{y \sim P_{\text{Mask}_{\text{GT}}}}[D(y, x)] \tag{5.15}$$

Where:

Mask_{GT} - A batch of ground-truth masks

$G(\bullet)$ - Generator function to generate masks from CT images

$D(\bullet)$ - Discriminator function to score the pair of masks and their corresponding CT images

$\mathbb{E}(\bullet)$ - Mean value of a distribution

P_z - Distribution of CT images in a batch

$P_{Mask_{GT}}$ - Distribution of ground truth masks in a batch

5.2.5 Regression Discriminator

The design of the regression discriminator, which will be detailed in this subsection, differs from other discriminator architectures. This design is motivated by our goal to approximate the distance between the two distributions of the scores of the generated masks and the ground truth masks through a regression process. The whole model made of the generator and the discriminator will be denoted as GAN_{REG} (see Figure 5.7).

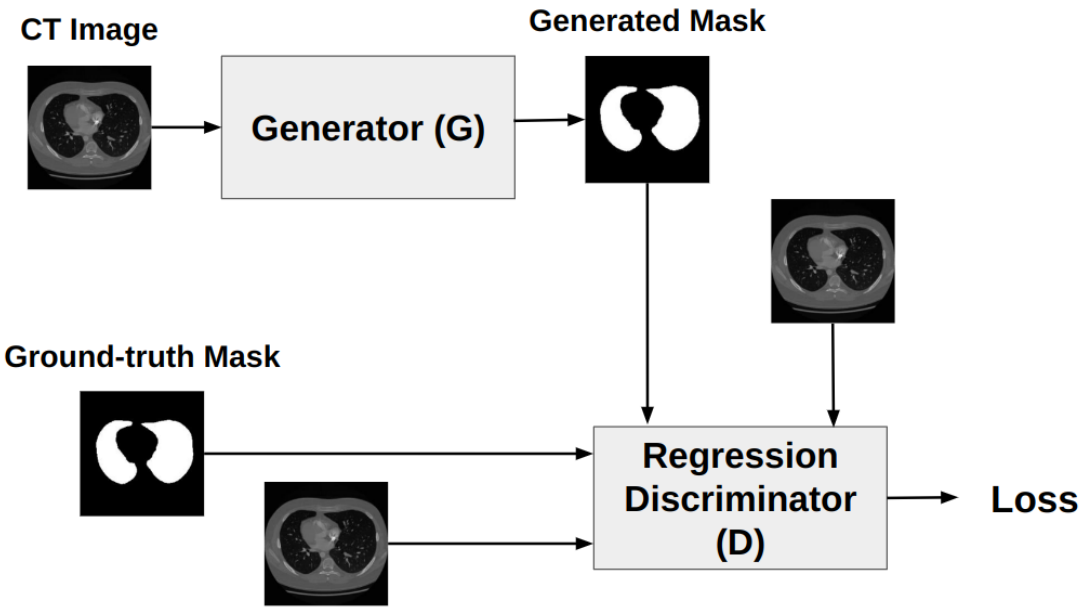

Figure 5.7: GAN$_{REG}$ architecture with a generator (G) and regression discriminator (D)

Architecture

The regression discriminator's architecture consists of two branches: one takes the generated mask as input, and the other takes the corresponding ground truth mask as input. Both masks undergo the same number of convolution operations until reaching a point where the extracted features from both branches are concatenated along the channels dimension. This discriminator is characterized by having more layers in both branches, indicating that lower-level features are learned before reaching the concatenation point. This allows more detailed information extraction from both masks.

The feature maps resulting from this concatenation go through additional layers for further processing. The result of the discriminator is an approximation of the distance between the generated mask and the ground truth mask.

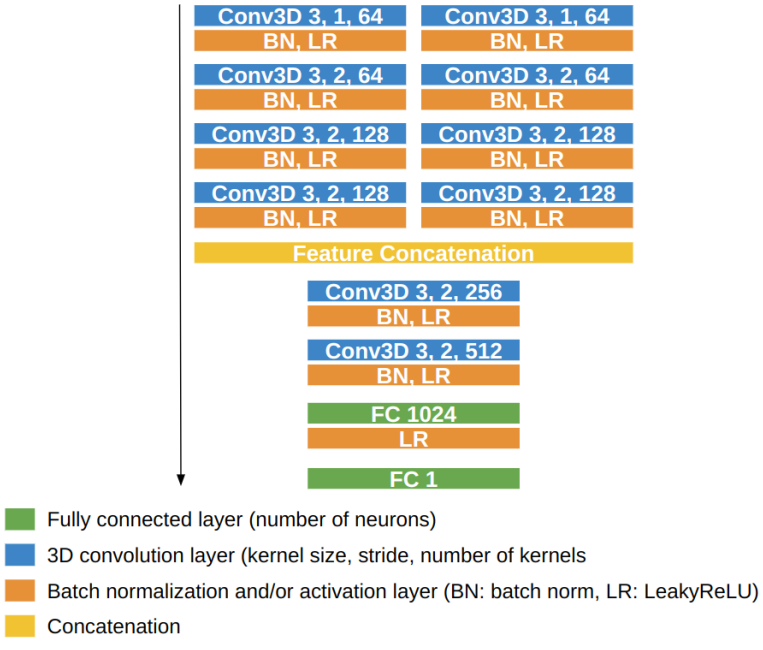

Figure 5.8: Regression discriminator architecture

Loss Function

In this context, the loss function for the generator remains the same as the loss function employed by the generator in GAN_B. The only change is due to the change of the inputs of the discriminator, which is also considered in the generator's loss. Consequently, the generator loss is expressed as 5.16.

When it comes to the discriminator, the principle is to minimize the mean of the discriminator's output. The output is an approximation of the distance between the two distributions of generated masks and ground truth masks. This approximation is a regression process which justify the name of the discriminator "regression discriminator".

$$\mathcal{L}_G = \mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x \sim P_z}[D(G(x), y)] \tag{5.16}$$

$$\mathcal{L}_D = \mathbb{E}_{x \sim P_z,\ y \sim P_{\text{Mask}_{\text{GT}}}}[D(G(x), y)] \tag{5.17}$$

Where:

Mask_{GT} - A batch of ground-truth masks

$G(\bullet)$ - Generator function to generate masks from CT images

$D(\bullet)$ - Discriminator function to score the pair of masks and their corresponding CT images

$\mathbb{E}(\bullet)$ - Mean value of a distribution

P_z - Distribution of CT images in a batch

$P_{Mask_{GT}}$ - Distribution of ground truth masks in a batch

5.3 Training and Results

Training

For our training process, we use Adam optimizer [16], as an optimization algorithm. We train our models using a batch size of 8. Hyperparameters are given as follows: a learning rate of 10^{-3}, a momentum of 0.9, weight decay set at 10^{-5}, and a total of 50 training epochs.

The training hyperparameters were used for all our models: the Base Model, GAN_B, GAN_P, GAN_{EF}, GAN_{LF}, GAN_{REG}.

For the training data, we used the dataset described in Chapter 3, after being preprocessed. From an initial set of 83 pairs of CT images and masks dedicated for training and validation, we reserved 32 pairs for validation.

The training algorithms for our base model, both when trained alone and within a GAN setup, are provided in Algorithms 1 and 2, respectively.

During testing, we observe the evolution of losses and validation metrics, which are also used for testing later, as we will see in the next subsection, during training for all our experiments. The curves are shown respectively in Figure 5.9 and Figure 5.10.

We save model weights each time these weights make the model perform better over a sequence of epochs. In other words, when the metrics remain above the moving average of metrics.

Algorithm 1: Base Model (SegResNet) Training for 3D Lung Segmentation

Input : CT Images (P_z), Ground truth masks (Mask_{GT})
Output: Trained generator G

for *epoch* $\leftarrow 1$ **to** N_{epochs} **do**
  Sample a batch of CT images and their ground truth masks $\{(x_1, y_1), (x_2, y_2), \ldots, (x_B, y_B)\}$ from (P_z, Mask_{GT});
  Generate a batch of segmentation masks $\{z_1, z_2, \ldots, z_B\}$ from $\{x_1, x_2, \ldots, x_B\}$ using G;
  Calculate generator loss $\mathcal{L}_G$ using the ground truth and generated masks;
  Update generator weights: $G \leftarrow G - \alpha \nabla_G \mathcal{L}_G$;

Algorithm 2: GAN Training Algorithm for 3D Lung Segmentation

Input : CT Images (P_z), Ground truth masks (Mask_{GT})
Output: Trained generator G and discriminator D

for *epoch* $\leftarrow 1$ **to** N_{epochs} **do**
  Sample a batch of CT images and their ground truth masks $\{(x_1, y_1), (x_2, y_2) \ldots, (x_B, y_B)\}$ from (P_z, Mask_{GT});
  Generate a batch of segmentation masks $\{z_1, z_2, \ldots, z_B\}$ from $\{x_1, x_2, \ldots, x_B\}$ using G;
  Calculate discriminator loss $\mathcal{L}_D$ using ground truth and generated masks;
  Update discriminator weights: $D \leftarrow D - \alpha \nabla_D \mathcal{L}_D$;
  Calculate generator loss $\mathcal{L}_G$ using the ground truth and generated masks;
  Update generator weights: $G \leftarrow G - \alpha \nabla_G \mathcal{L}_G$;

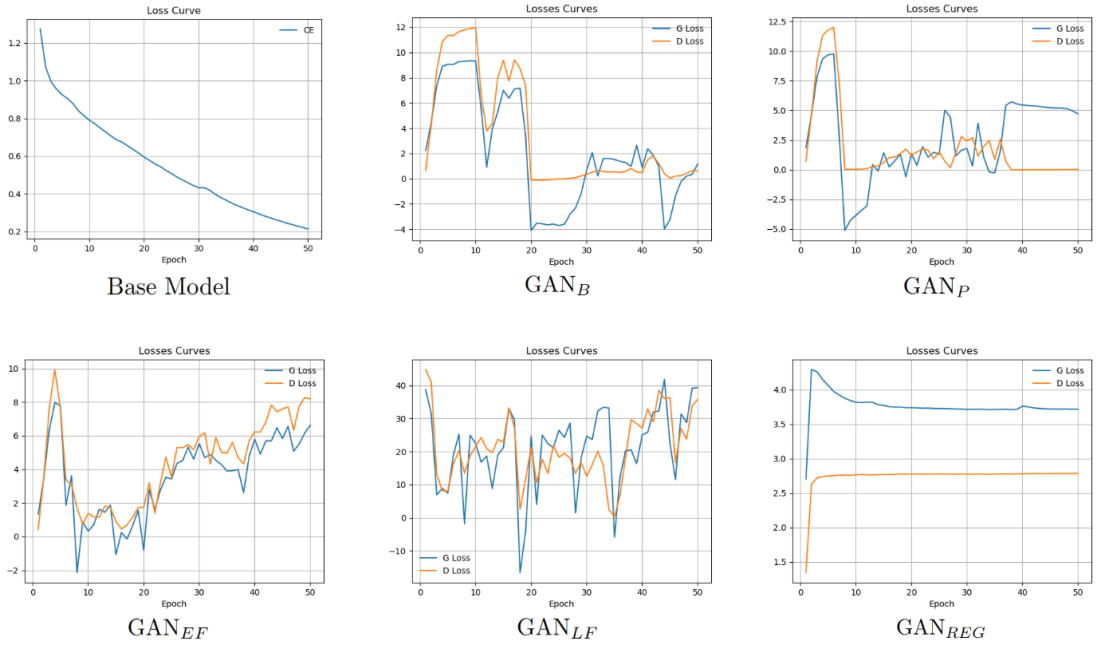

Figure 5.9: Evolution of loss functions over epochs during model training

Models	MeanDSC	MeanIoU	MeanHD
Base Model	0.9829	0.9670	17.83
GAN_B	0.9819	0.9651	28.01
GAN_P	0.9745	0.9731	27.59
GAN_{EF}	0.9876	0.9757	**14.58**
GAN_{LF}	**0.9906**	**0.9815**	21.43
GAN_{REG}	0.9901	0.9807	20.85

Table 5.1: Computed MeanIoU, MeanDSC and MeanHD on test data for our six models

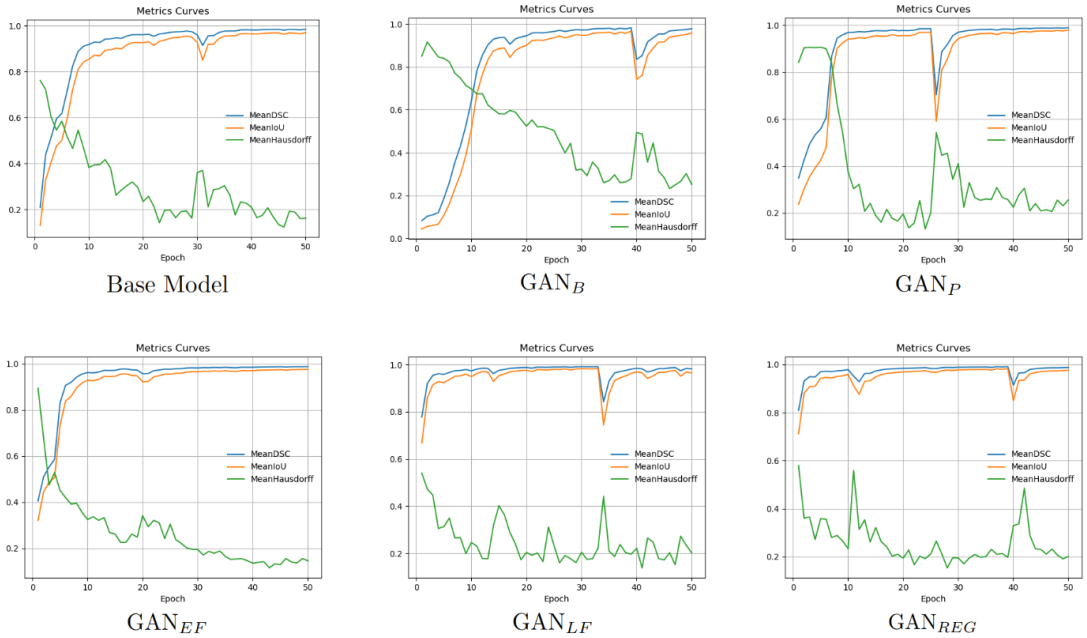

Figure 5.10: Evolution of metrics (MeanDSC, MeanIoU and scaled MeanHD) over epochs during model training

Testing

Testing the model's performance is conducted using the generator alone, as it is responsible for generating segmentation masks, whether it was trained as a standalone model or in an adversarial training setup. At this stage, we no longer consider the discriminator for GAN models.

Testing is made on the 80 pairs of CT images and their corresponding masks. The model's performance is quantitatively assessed using the IoU metric and the DSC metric, as explained in 2.3.1. However, since the generated mask consists of 3 channels and has dimensions $[1, 3, 128, 128, 128]$, where 1 is the batch size, 3 is the number of channels, and 128 represents the size along the X, Y, and Z axes, we compute the DSC, the IoU and the HD for each channel of the generated mask compared to the ground truth mask, which gives a vector of length 3 for each metric. Subsequently, we calculate the mean of eahc of this vectors, which we refer to as MeanIoU, MeanDSC and MeanHD metrics.

In the testing step, we computed the metrics MeanIoU and MeanDSC, and the results are listed in Table 5.1.

Discussion

The models perform well quantitatively, as shows the metrics 5.1, where the performance is better for different GANs compared to the Base Model alone and also in the superposition

of the CT images and the generated lung segmentation masks when the gastric bubble is positioned far from the lung, as shows Figures 5.11, 5.12, 5.13. However, some models often fails to distinguish between the gastric bubble voxels and the lung voxels on CT images where the gastric bubble is very close to the lung, as it is the case for some CT images (see Figure 5.14).

The gastric bubble is visible when viewed along the Y-axis. We will focus exclusively on slice 55, which is sliced along the Z-axis and viewed along the Y-axis.

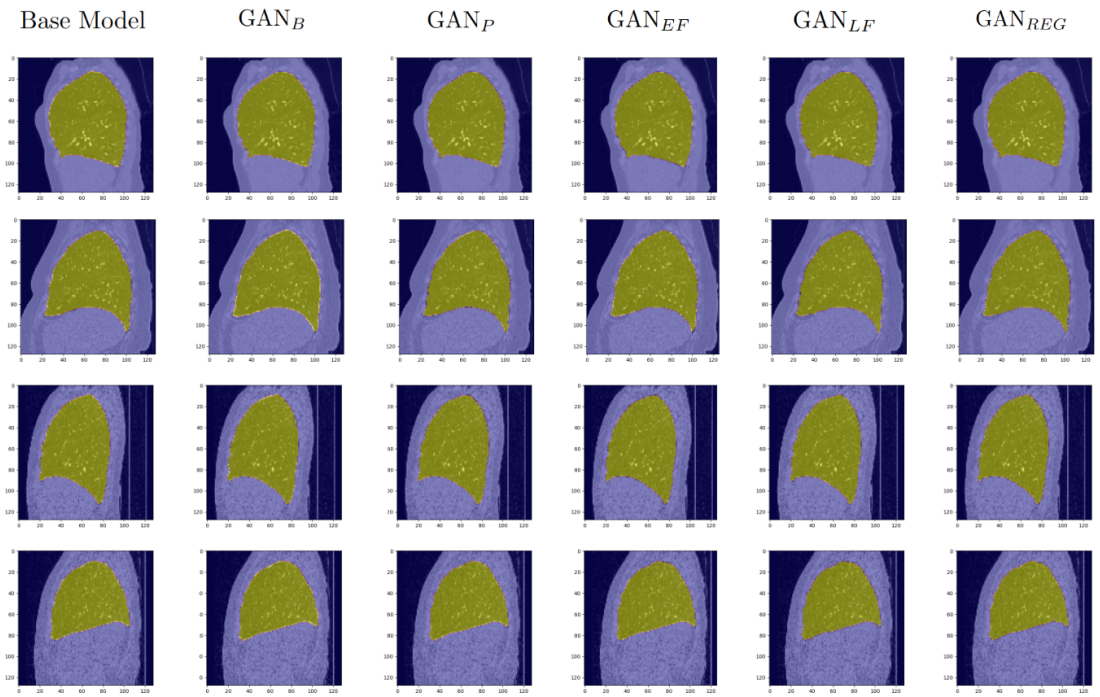

Figure 5.11: Slice 55 of the segmentation masks (yellow) generated on individual CT images (purple) as viewed from the X-axis perspective. Each column corresponds to a model, and each row corresponds to a patient

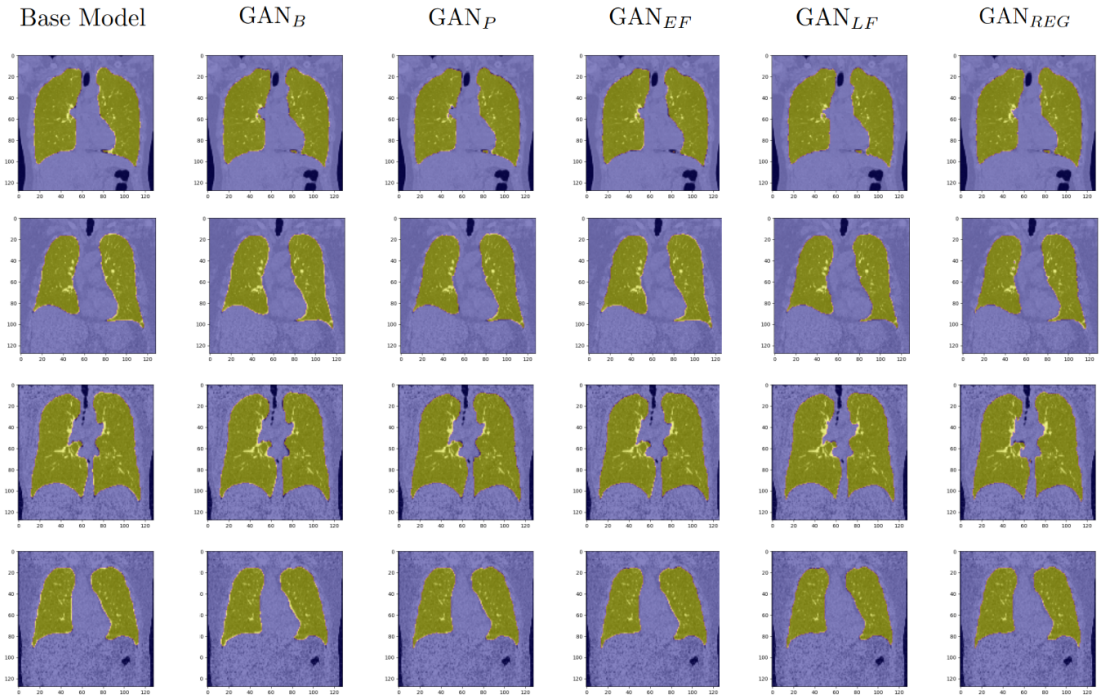

Figure 5.12: Slice 55 of the segmentation masks (yellow) generated on individual CT images (purple) as viewed from the Y-axis perspective. Each column corresponds to a model, and each row corresponds to a patient

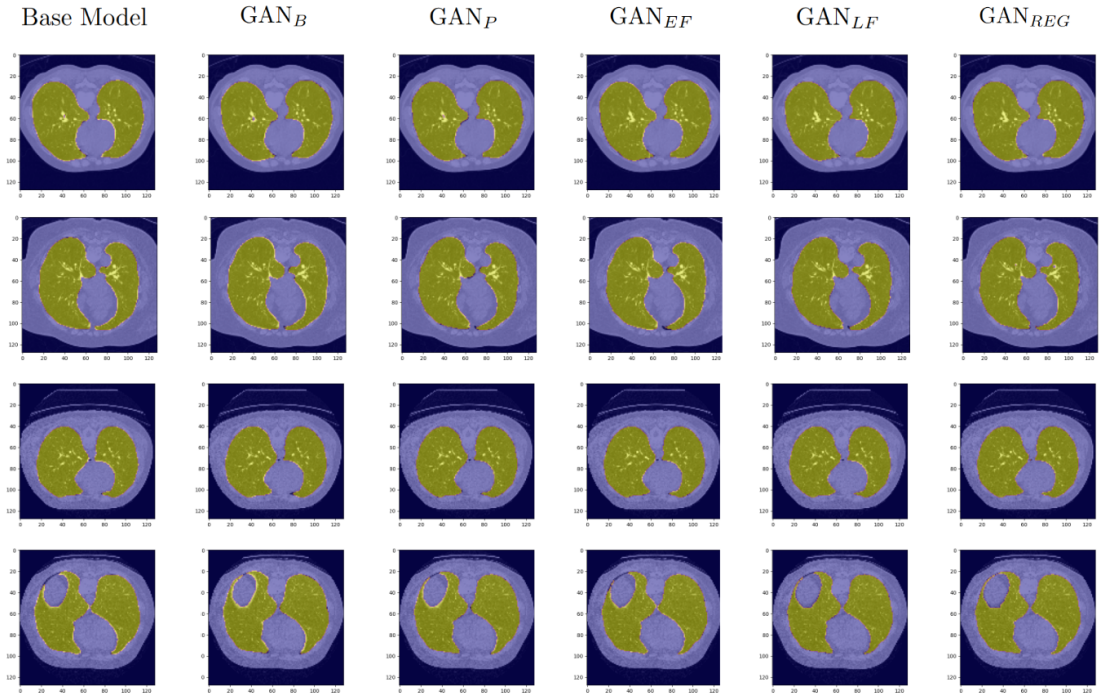

Figure 5.13: Slice 55 of the segmentation masks (yellow) generated on individual CT images (purple) as viewed from the Z-axis perspective. Each column corresponds to a model, and each row corresponds to a patient

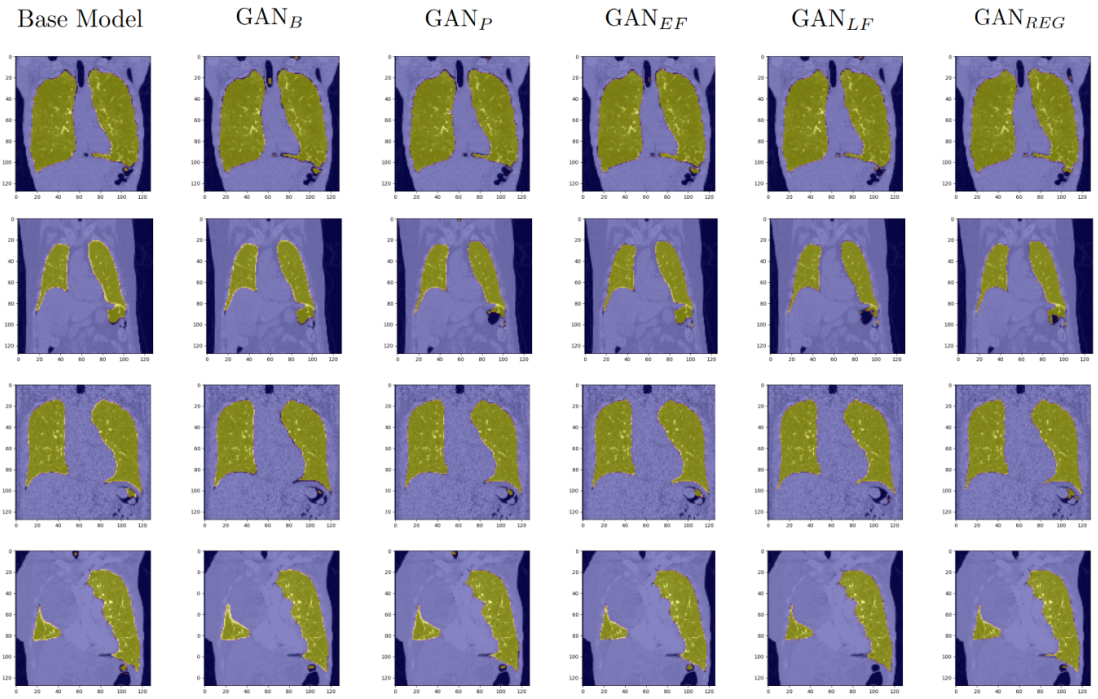

Figure 5.14: Slice 55 of the segmentation masks (in yellow) generated from individual CT images (in purple), viewed from the Y-axis perspective. Each column represents a model, and each row represents a patient. Notably, for all four patients, the gastric bubble is very close to the lung

The idea of enhancing the base model's performance through adversarial training does not effectively address this issue, some models fail to generate a lung segmentation mask that doesn't include the gastric bubble pixels. Consequently, we opted for a supplementary solution to reduce the true positives. This method will be detailed in the next section.

5.4 Ensemble Method

This method consists of performing an "ensembling": based on the segmentation masks generated by different models, we assign a score to indicate the likelihood of a voxel belonging to a particular class. The objective is to have the models participating in this scoring process "vote" on whether a voxel is part of the gastric bubble or not.

Ensemble methods, are modern approaches in machine learning, that excel in enhancing prediction accuracy by combining the collective predictions of diverse models. Instead of relying on a single model with its limitations, ensemble methods use the strength of multiple models. By merging their outputs, they improve robustness and generality. Different classical models are based on ensemble methods, including the Random Forest classifier [13], Gradient Boosting Machines [7], AdaBoost [6], and others.

These methods consistently demonstrate superior performance across various tasks compared to the use of a single model. For instance, when comparing a single decision tree to a Random Forest classifier for a classification task, it becomes evident that the Random Forest classifier offers better performance and better decision-making.

Scoring Pipeline

The scoring pipeline takes two parameters as input: the pretrained models where N_{models} is the number of models and a contibution rate denoted as R_c, between 0 to 1. This contribution

rate is used to compute the thershold as follows: $T = R_c \times N_{models}$.

We generate segmentation masks for all the test CT scans. These masks are gathered from the various models where each generated mask is binary. For each CT image we sum the generated masks issued by each model (see 5.18), as a result we have a resulting 128×128×128 matrix with coefficients between 0 and N_{models}. If a voxel's value surpasses the threshold T, the scoring model determines that the voxel belongs to the lung. On the other hand, if the value is below the threshold, the voxel is classified as not part of the lung (see 5.19).

The scoring pipeline is designed as shown in Figure 5.15.

$$S^l_{ijk} = \sum_{m \in \mathcal{M}} (A_{ijk})^l_m \tag{5.18}$$

$$O^l_{ijk} = \begin{cases} 1 & \text{if } S^l_{ijk} \geq T \\ 0 & \text{else} \end{cases} \tag{5.19}$$

O^l_{ijk} – Voxel value in position $i, j,$ and k, along axes X, Y, and Z, respectively, in the generated mask from CT image l after scoring

$(A_{ijk})^l_m$ – Voxel value in position $i, j,$ and k, along axes X, Y, and Z, respectively, in the generated mask from CT image l through model m

S^l_{ijk} – Score of generated mask from CT image l ($S^l_{ijk} \in [0, N_{\text{models}}]$)

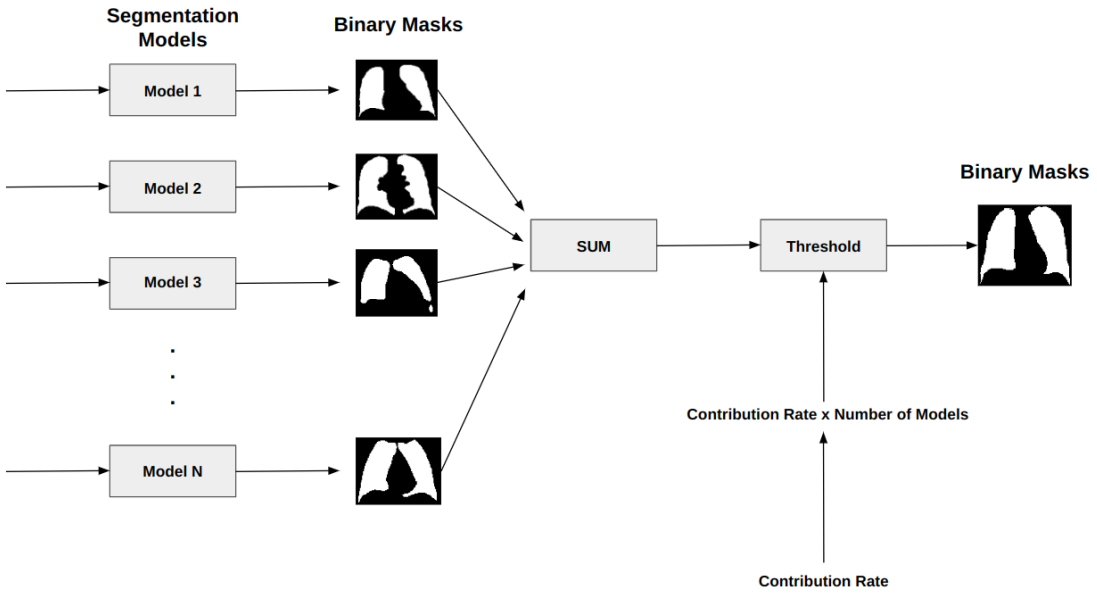

Figure 5.15: Architecture of the scoring pipeline

Application

We applied this scoring pipeline using our six pretrained models: the Base Model, GAN_B, GAN_P, GAN_{EF}, GAN_{LF}, and GAN_{REG}, resulting in $N\text{models} = 6$. We used a contribution

Models	MeanDSC	MeanIoU	MeanHD
Scoring Pipeline	0.9859	0.9723	**11.66**

Table 5.2: Computed MeanIoU, MeanDSC and MeanHD on test data after scoring

rate $R_c = 0.8$, which means that a voxel is considered part of the lung if it is identified as such by at least 80% of the models.

After scoring on the preprocessed test data, on which the models were previously trained, we improved the MeanHD which is closely related to the contour difference between the ground truth masks and the generated masks, as we can see by comparing the MeanHD in Table 5.1 and Table 5.1.

The decrease in MeanDSC and MeanIoU may be due to the disappearance of voxels along the lung's contour after scoring. This occurs due to the models' uncertainty on whether these voxels are in the segmentation mask. However, this typically does not pose significant issues if the error is limited to very few voxels located near the contours.

In Figure , it is evident that the generated masks after scoring significantly reduced the size of the gastric bubble, confirming the quantitative improvement in MeanHD (see Table 5.2).

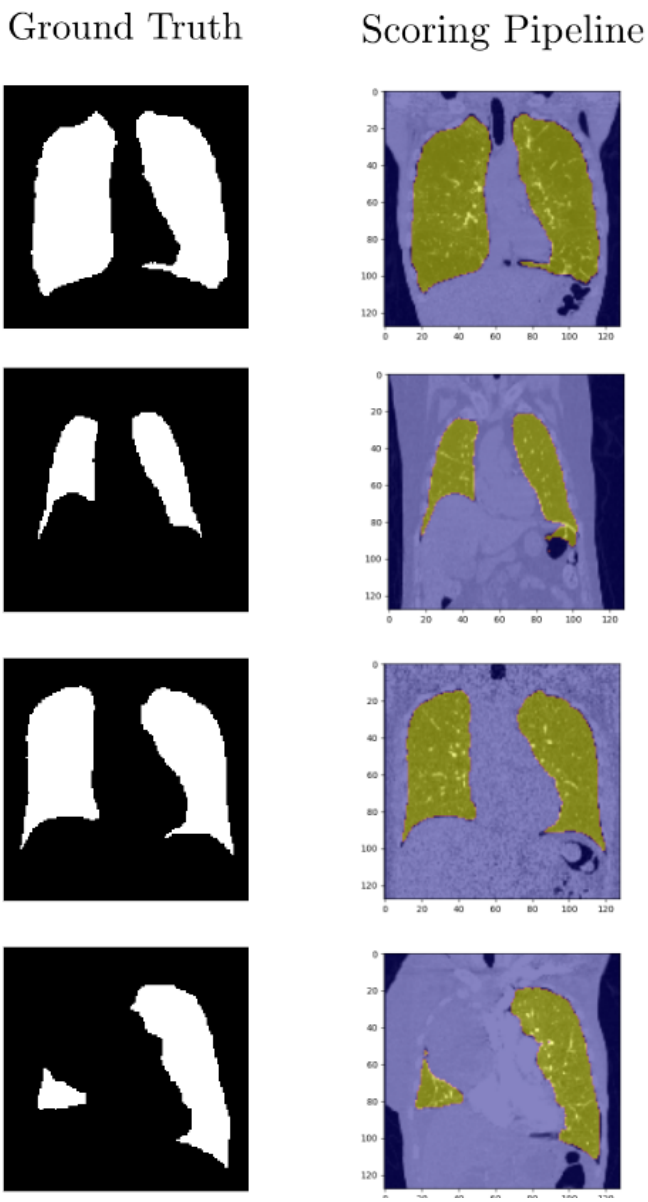

Figure 5.16: Slice 55 of the segmentation masks (in yellow) generated from individual CT images (in purple), as viewed from the Y-axis perspective (right column). Slice 55 of the ground truth mask, also viewed from the Y-axis perspective (left column). Each row corresponds to the same patient, as shown in Figure 5.14

This ensemble method may still have some limitations, particularly when the gastric bubble is not significantly reduced, especially when it is located directly under the lung, as illustrated in Figure 5.17.

While this method aids in reducing false positives in many instances, its accuracy could potentially be enhanced with the inclusion of additional pretrained models. We observe improvements using our current models, but using more models with diverse architectures and training parameters has the potential to yield substantial enhancements.

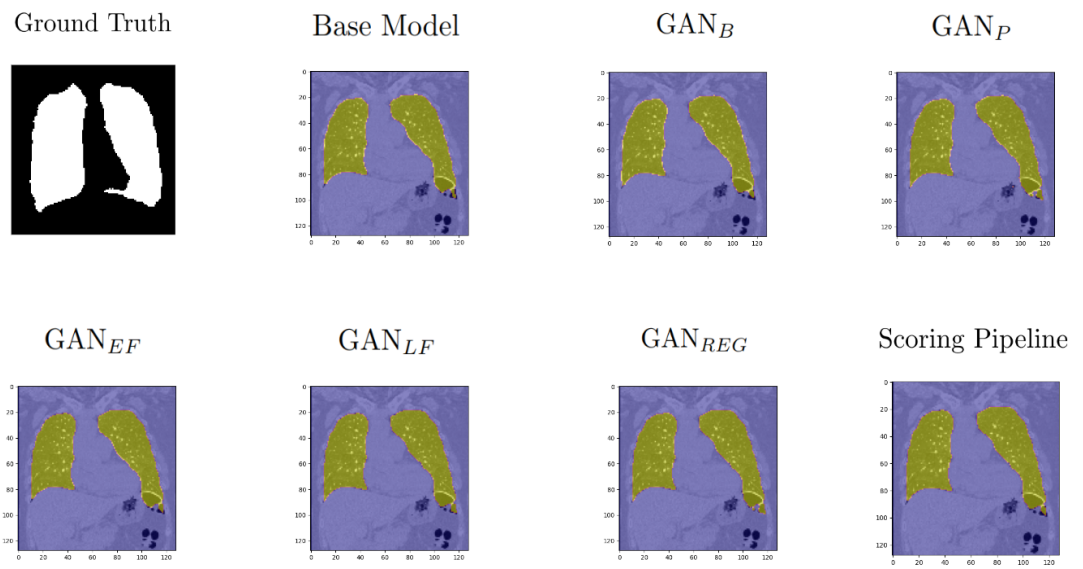

Figure 5.17: An example where the gastric bubble is very close to the lung. The proposed ensemble method fail to reduce false positives related to the gastric bubble

5.5 Customized Loss Method

In this section, we will present a secondary approach aimed at reducing false positives in lung segmentation. These false positives mainly consist of voxels incorrectly classified as part of the lung by the model.

Our idea involves implementing a segmentation loss that incorporate the Hausdorff loss [15] besides to the cross-entropy. Specifically, we maintain the cross-entropy ($\mathbb{CE}$) with a weight factor, w_{CE}, and introduce the Hausdorff loss ($\mathbb{HD}$) with a weight factor, w_{HD}, both ranging between 0 and 1.

The goal is to find an appropriate weighting for $\mathbb{CE}$ and $\mathbb{HD}$ that optimally balances their contributions.

We applied this method exclusively to a basic discriminator to assess whether it would lead to improvements in terms of reducing false positives in the generated lung segmentation mask, particularly in the context of gastric bubble reduction. The model will be denoted GAN$'_B$

Loss Design

As described above the customized loss function for the generator and the normal loss for the discriminator of GAN$'_B$ are defined, respectively in 5.20, 5.21.

$$\mathcal{L}_G = w_{CE}\mathbb{CE}(G(x), \text{Mask}_{\text{GT}}) + w_{HD}\mathbb{HD}(G(x), \text{Mask}_{\text{GT}}) - \mathbb{E}_{x\sim P_z}[D(G(x), y)] \tag{5.20}$$

$$\mathcal{L}_D = \mathbb{E}_{x\sim P_z,\ y\sim P_{\text{Mask}_{\text{GT}}}}[D(G(x), y)] \tag{5.21}$$

Experiment and Results

We conducted a single experiment due to time constraints. We followed the same training process and used the same training parameters outlined in Section 5.3. The only modification is

Models	MeanDSC	MeanIoU	MeanHD
GAN'_B	0.9838	0.9684	**12.48**

Table 5.3: Computed MeanIoU, MeanDSC and MeanHD on test data of GAN'_B

in terms of the generator loss, which differs from the GAN_B described earlier, as it incorporates segmentation.

n this experiment, we assigned a weight of 0.4 to w_{CE} and 0.6 to w_{HD}, nearly balancing the contributions of the Hausdorff loss and the cross-entropy loss. This allocation targets the central regions of the segmentation mask at 40% and the segmentation mask contours at 60%.

As a result, we obtained the metrics after testing on the preprocessed test data, as presented in Table 5.3.

The MeanHD is smaller than all the MeanHD of our six models presented in Table 5.1. This confirms an improvement in terms of precision in approximating ground truth masks, particularly when it comes to adjusting the contours.

The qualitative results in Figure 5.18 confirm this improvement. It is clear evident that the presence of gastric bubbles has been significantly reduced compared to the outcomes of the six previous models.

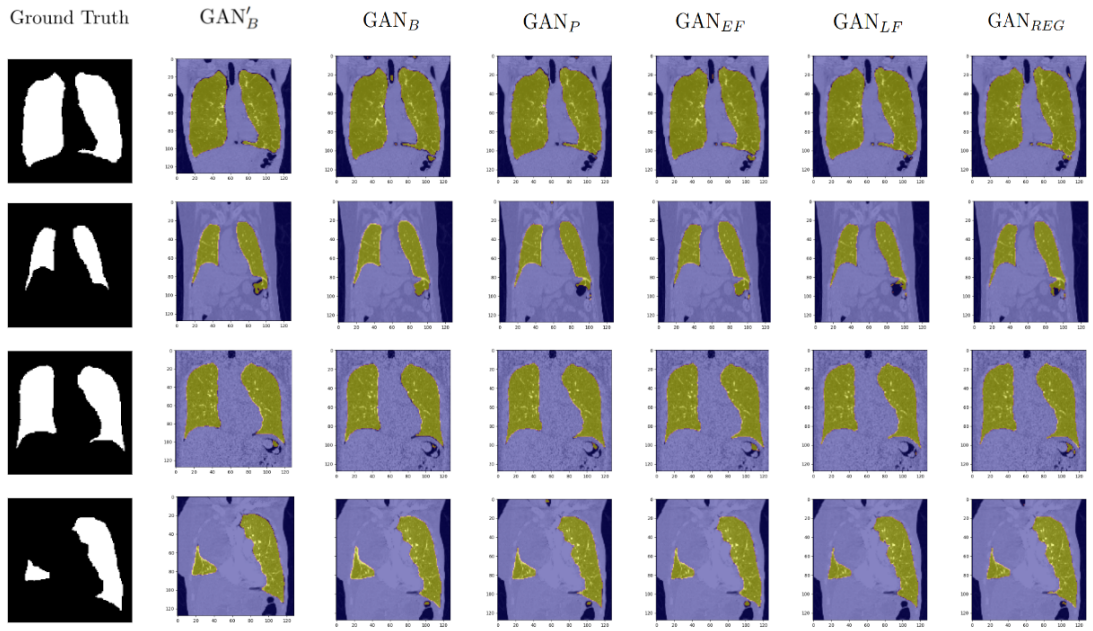

Figure 5.18: Slice 55 of the segmentation masks (in yellow) generated from individual CT images (in purple), viewed from the Y-axis perspective. Each column represents a model, and each row represents a patient. Notably, for all four patients, the gastric bubble is very close to the lung

Chapter 6

Conclusion and Future Work

In this section we will estabilish a conclusion of all the work presented above and we will open new perspectives of the work that could have been done to improve the results.

6.1 Conclusion

During my internship, my work focus was around enhancing 3D lung segmentation from CT scans, a very important step in Median Technologies' lung cancer screening workflow. The challenge was to accurately segment the lungs, particularly when the gastric bubble was in close proximity, often resulting in mispredicting gastric bubble voxels as a part of the lung. To tackle this issue, we employed various architectures inspired by WGAN principles. The base model, SegResNet, was trained both alone and within different GAN configurations, each using different discriminator architecture. The goal was to use the discriminator's learned prior knowledge to refine the generated masks.

Through adversarial training and the use of specialized loss functions, some models showed improved lung segmentation precision. This improvement can be seen quantitatively through the IoU and DSC metrics, which showed enhanced performance compared to base model. Moreover, visually, the overlap with ground truth masks also this enhancement in some models.

However, in cases where the lung was in very close to the gastric bubble, the models did not exhibit significant improvement. To mitigate false positives in the segmentation mask, we implemented an ensemble scoring pipeline that combined predictions from multiple models. This approach relied on the models' "votes" to reduce incorrect labeling of gastric bubble voxels.

We made also some preliminary efforts to implement customized loss functions for the generator with the goal of improving segmentation and reducing the inclusion of gastric bubble areas in the lung segmentation mask.

This internship project's goal is the enhancement in the accuracy of lung cancer screening, primarily through robust lung segmentation, which is important for lesion detection in next analysis steps.

6.2 Future Work

For future work, we can explore the impact of adjusting the weights in the customized loss of GAN'_B. Given more time, we could inspect this aspect in more detail.

We suggest also to particularly inspect in details the latent space where the compressed information from CT scans is decoded to generate segmentation masks.

The issue of over-segmentation, where the model includes the gastric bubble as part of the lung, may be linked to the learned latent space resulting from the encoding process of the SegResNet model. In scenarios where the majority of patients have a gastric bubble positioned far from the lung, the latent representation tends to adapt to this standard lung representation. As a result, the latent representation of CT scans tends to center around this mean lung representation, which leads to the misclassification of certain voxels that are close to the lung and similar to lung tissue as part of the lung. Consequently, the gastric bubble or a portion of it is incorrectly classified as part of the lung.

This optimization technique consisted of training a model to predict one of the segmentation evaluation metrics using the feature maps derived from the forward propagation of CT image through the encoder. In other words, we wanted to perform regression to predict the metric from the latent space feature maps.

The next step consisted of using the regression model to predict the evaluation score and perform a gradient descent from this prediction to optimize the latent space feature maps of the CT images directly. This optimization would act as a post-processing method and keep the encoder and decoder parameters unchanged, and add only prior knowledge to the latent features to enhance the generated mask.

We started implementing this method towards the end of the internship. The primary challenge encountered was related to the regression task, where the predicted metric showed small variations between different CT scans. Additionally, predicting the correlation between the latent space and the metric showed to be very difficult, due to the highly compressed information in the latent space feature maps. Time constraints limited our ability to work deeper on this optimization technique, but it could be a very insightful and accurate method to work on for further improvement of the segmentation outcomes.

Bibliography

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

[2] Henry Blumberg. "Hausdorff's Grundzüge der Mengenlehre". In: (1920).

[3] Özgün Çiçek et al. "3D U-Net: learning dense volumetric segmentation from sparse annotation". In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*. Springer. 2016, pp. 424–432.

[4] Jean Cousty et al. "Watershed cuts: Minimum spanning forests and the drop of water principle". In: *IEEE transactions on pattern analysis and machine intelligence* 31.8 (2008), pp. 1362–1374.

[5] Lee R Dice. "Measures of the amount of ecologic association between species". In: *Ecology* 26.3 (1945), pp. 297–302.

[6] Yoav Freund and Robert E Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of computer and system sciences* 55.1 (1997), pp. 119–139.

[7] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics* (2001), pp. 1189–1232.

[8] Kunihiko Fukushima. "Cognitron: A self-organizing multilayered neural network". In: *Biological cybernetics* 20.3-4 (1975), pp. 121–136.

[9] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[10] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[11] Mohammad Havaei et al. "Brain tumor segmentation with deep neural networks". In: *Medical image analysis* 35 (2017), pp. 18–31.

[12] Kaiming He et al. "Identity mappings in deep residual networks". In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer. 2016, pp. 630–645.

[13] Tin Kam Ho. "Random decision forests". In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE. 1995, pp. 278–282.

[14] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

[15] Davood Karimi and Septimiu E Salcudean. "Reducing the hausdorff distance in medical image segmentation with convolutional neural networks". In: *IEEE Transactions on medical imaging* 39.2 (2019), pp. 499–513.

[16] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[17] Ralf Kohler. “A segmentation system based on thresholding”. In: *Computer Graphics and Image Processing* 15.4 (1981), pp. 319–338.

[18] Mark A Kramer. “Nonlinear principal component analysis using autoassociative neural networks”. In: *AIChE journal* 37.2 (1991), pp. 233–243.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).

[20] Lahouaoui Lalaoui and Tayeb Mohamadi. “A comparative study of image region-based segmentation algorithms”. In: *International Journal of Advanced Computer Science and Applications* 4.6 (2013).

[21] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[22] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. “Rectifier nonlinearities improve neural network acoustic models”. In: *Proc. icml*. Vol. 30. 1. Atlanta, GA. 2013, p. 3.

[23] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. “V-net: Fully convolutional neural networks for volumetric medical image segmentation”. In: *2016 fourth international conference on 3D vision (3DV)*. Ieee. 2016, pp. 565–571.

[24] Allan H Murphy. “The Finley affair: A signal event in the history of forecast verification”. In: *Weather and forecasting* 11.1 (1996), pp. 3–20.

[25] Andriy Myronenko. “3D MRI brain tumor segmentation using autoencoder regularization”. In: *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4*. Springer. 2019, pp. 311–320.

[26] 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*. Springer. 2015, pp. 234–241.

[27] Arnaud Arindra Adiyoso Setio et al. “Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: the LUNA16 challenge”. In: *Medical image analysis* 42 (2017), pp. 1–13.

[28] Claude Elwood Shannon. “A mathematical theory of communication”. In: *The Bell system technical journal* 27.3 (1948), pp. 379–423.

[29] Thorvald Sorensen. “A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons”. In: *Biologiske skrifter* 5 (1948), pp. 1–34.

[30] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[31] Jiaxing Tan et al. “LGAN: Lung segmentation in CT scans using generative adversarial network”. In: *Computerized Medical Imaging and Graphics* 87 (2021), p. 101817.

[32] Leonid Nisonovich Vaserstein. “Markov processes over denumerable products of spaces, describing large systems of automata”. In: *Problemy Peredachi Informatsii* 5.3 (1969), pp. 64–72.

[33] Yuxin Wu and Kaiming He. “Group normalization”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19.

[34] Zongwei Zhou et al. “Unet++: A nested u-net architecture for medical image segmentation”. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*. Springer. 2018, pp. 3–11.

[35] Djemel Ziou, Salvatore Tabbone, et al. “Edge detection techniques-an overview”. In: *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii* 8 (1998), pp. 537–559.