



**POLITECNICO**  
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE  
E DELL'INFORMAZIONE

# Automated Left Ventricle Segmentation in Cardiac CT Images using Deep Learning Techniques

TESI DI LAUREA MAGISTRALE IN  
COMPUTER SCIENCE AND ENGINEERING - INGEGNERIA INFORMATICA

Author: **Juan F. Calderon**

Student ID: 944557

Advisor: Prof. Valentina Corino

Co-advisors: Francesca Lo Iacono

Academic Year: 2022-23



# Abstract

The segmentation of the left ventricle (LV) in cardiac computed tomography (CT) images is an important task for the diagnosis and treatment of cardiovascular diseases. Manual segmentation is time-consuming and subject to inter- and intra-observer variability. Therefore, the development of accurate and automated segmentation methods is crucial. In this thesis, we investigate the use of deep learning models for the segmentation of the LV in cardiac CT images.

We evaluated three different deep learning models: U-Net, 2.5D U-Net, and U-Net with LSTM. The models are trained and evaluated using a dataset of 85 patients, which are manually segmented by an expert radiologist. The segmentation performance of the models is evaluated using quantitative metrics such as the Dice coefficient and Intersection over Union coefficient, as well as visual inspection of the segmentation results. Our results show that the 2.5D U-Net model achieves the best segmentation performance. The qualitative evaluation of the segmentation results using visual inspection of the images and corresponding masks reveals that the 2.5D U-Net model produces the most accurate segmentations overall, with the fewest errors and artifacts.

Overall, our results demonstrate the potential of deep learning models for the segmentation of the left ventricle in cardiac CT images. The 2.5D U-Net model, in particular, shows superior performance, with high accuracy and few errors or artifacts.

**Keywords:** Deep Learning, UNet, Image Segmentation, CT Scan, Left Ventricle



# Abstract in lingua italiana

La segmentazione del ventricolo sinistro (LV) nelle immagini di tomografia computerizzata (TC) cardiaca è un'importante attività per la diagnosi e il trattamento delle malattie cardiovascolari. La segmentazione manuale richiede molto tempo ed è soggetta a variabilità inter- e intra-osservatore. Pertanto, lo sviluppo di metodi di segmentazione accurati e automatizzati è cruciale. In questa tesi, abbiamo studiato l'uso di modelli di deep learning per la segmentazione del LV nelle immagini TC cardiache.

Abbiamo valutato tre diversi modelli di deep learning: U-Net, U-Net 2.5D e U-Net con LSTM. I modelli sono stati addestrati e valutati utilizzando un dataset di 85 pazienti, i cui LV sono stati manualmente segmentati da un radiologo esperto. Le prestazioni di segmentazione dei modelli sono state valutate utilizzando metriche quantitative come il coefficiente di Dice e il coefficiente di Intersezione sull'Unione, nonché l'ispezione visiva dei risultati di segmentazione. I nostri risultati mostrano che il modello 2.5D U-Net raggiunge le migliori prestazioni di segmentazione. La valutazione qualitativa dei risultati di segmentazione utilizzando l'ispezione visiva delle immagini e delle maschere corrispondenti rivela che il modello 2.5D U-Net produce le segmentazioni più accurate in generale, con meno errori e artefatti.

Complessivamente, i nostri risultati dimostrano il potenziale dei modelli di deep learning per la segmentazione del ventricolo sinistro nelle immagini TC cardiache. In particolare, il modello 2.5D U-Net mostra prestazioni superiori, con elevata precisione e pochi errori o artefatti.

**Parole chiave:** Deep Learning, UNet, Image Segmentation, CT Scan, Ventricolo sinistro



# Contents

<b>Abstract</b>	<b>i</b>
<b>Abstract in lingua italiana</b>	<b>iii</b>
<b>Contents</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Deep learning . . . . .	1
1.2 Convolutions . . . . .	2
1.2.1 Convolution operator . . . . .	3
1.2.2 Pooling layer . . . . .	5
1.2.3 Non linear activation function . . . . .	6
1.2.4 Learning rate . . . . .	7
1.2.5 Optimization . . . . .	8
1.3 CT scan images . . . . .	8
1.4 Image segmentation . . . . .	9
1.4.1 UNet . . . . .	10
1.5 RNN . . . . .	12
1.5.1 LSTM . . . . .	13
1.5.2 Motivation of the study . . . . .	14
<b>2 Materials and Methods</b>	<b>15</b>
2.1 Population . . . . .	15
2.2 CT scan protocol . . . . .	17
2.3 . . . . .	18
2.4 Image Preprocessing . . . . .	18
2.4.1 CT Slice Order and Dataset Cleaning . . . . .	18
2.4.2 Hounsfield unit transformation (HU) . . . . .	19
2.4.3 Windowing . . . . .	20

2.4.4	Data Augmentation Techniques . . . . .	21
2.5	Evaluation Metrics in Segmentation . . . . .	22
2.5.1	Loss Functions for Image Segmentation . . . . .	24
2.6	Models . . . . .	25
2.6.1	UNet . . . . .	25
2.6.2	UNet 2.5D . . . . .	27
2.6.3	UNet-LSTM . . . . .	28
2.7	GUI for Mask Visualization . . . . .	30
<b>3</b>	<b>Results</b>	<b>33</b>
3.1	Post processing . . . . .	33
3.2	Validation . . . . .	34
3.2.1	UNet . . . . .	34
3.2.2	UNet 2.5D . . . . .	35
3.2.3	UNet-LSTM . . . . .	36
3.2.4	Model comparison . . . . .	37
3.2.5	Visual inspection . . . . .	39
3.3	Test set . . . . .	41
3.3.1	Global metrics . . . . .	42
3.3.2	Visual inspection . . . . .	44
<b>4</b>	<b>Discussion</b>	<b>51</b>
4.1	Literature review . . . . .	51
4.2	UNet 2.5D . . . . .	53
<b>5</b>	<b>Conclusions and future developments</b>	<b>55</b>
5.1	Strenghts . . . . .	55
5.2	Limitations . . . . .	56
5.3	Future works . . . . .	56
	<b>Bibliography</b>	<b>59</b>
	<b>A Appendix A</b>	<b>63</b>
	<b>List of Figures</b>	<b>65</b>
	<b>List of Tables</b>	<b>69</b>





# 1 | Introduction

## 1.1. Deep learning

Deep learning (DL) algorithms have emerged as a prominent research area within the fields of machine learning and artificial intelligence [1]. These algorithms enable the analysis of large-scale data through the utilization of deep neural networks, which facilitate learning from experience in a manner akin to the human brain. This supervised learning approach allows for the extraction of features from extensive datasets without human intervention.

Traditional machine learning methods require human-designed features that are subsequently input into a predictive model. In contrast, DL algorithms employ end-to-end learning, enabling models to learn directly from examples and rectify past mistakes. The power of DL lies in its capacity to model highly nonlinear relationships between inputs and desired outputs by organizing multiple neurons into layers. This process yields various levels of abstract features that can be utilized for target detection, classification, or segmentation [2]. However, DL models necessitate large volumes of data, specialized GPUs for training, and longer training durations compared to classical machine learning approaches.

Artificial neural networks (ANNs) form the basis of prevalent DL methods. An ANN consists of numerous interconnected neurons that collectively constitute a deep neural network. For instance, when analyzing an image, each pixel is connected to all hidden neurons, with multiple layers employed to learn distinct types of features. Each neuron multiplies the preceding value by a weight, and the multiplied inputs are aggregated, with a bias coefficient added to the summation. The resulting value is then passed through a nonlinear activation function, which determines whether a neuron should transmit output to the subsequent layer's neurons. Initially, weights and biases are assigned randomly and adjusted during training using the backpropagation algorithm. The output layer produces a probability value.

Depending on the task at hand, an optimal neural network structure can be selected to best suit the specific task. The term architecture refers to the number of layers and

neurons within the neural network and their interconnections. In image analysis, the most prevalent architecture is the convolutional neural network (CNN), which has been extensively employed for image and object recognition and classification tasks [3] [4].

CNN demonstrate exceptional feature extraction capabilities, eliminating the need for manual image feature extraction or excessive preprocessing. By deriving high-level features from low-level ones, CNNs can extract hierarchical features. Consequently, CNNs have experienced considerable success in the domain of medical imaging diagnosis [50].

In summary, DL algorithms, primarily based on artificial neural networks, have become a significant area of research in machine learning and artificial intelligence. These algorithms enable the analysis of large-scale data by simulating the human brain's learning process and extracting features in an unsupervised manner. The strength of DL lies in its ability to model nonlinear relationships and generate abstract features for various applications.

## 1.2. Convolutions

In the field of image and video analysis, specialized neural networks (CNNs) are employed. While Fully Connected Networks excel in raw numerical data analysis, they are inadequate for image processing. There are crucial distinctions between numerical datasets and image datasets. Primarily, images and volumes are represented by 2D and 3D vectors composed of individual pixels or voxels, which can assume up to three values for colored images. This considerably expands the size of input data, rendering the notion of allocating a set of parameters to each single unit infeasible. Excessive connections would be necessary, and computations would become prohibitively expensive. Furthermore, spatial information is vital and must be accounted for in some manner. Treating each pixel (or voxel) as a distinct, isolated input entity would eliminate any global contextual information. Additionally, each element is closely connected to its surroundings, necessitating a thorough analysis that takes this into consideration.

Both CNNs and normal neural networks have a sequence of layers that transmit and modify information from input to output. Each layer consists of a set of neurons that apply a specific operation (such as a linear transformation or a nonlinear activation function) to the input data. The output of one layer becomes the input to the next layer, and this process continues until the final layer produces the model's output. The key distinction lies in the operation executed within each network layer, in this case, convolution. This mathematical technique significantly reduces the number of parameters within the network, thereby substantially enhancing computational efficiency. Concurrently, it enables the processing of images in their entirety, facilitating the learning of spatial relationships

between individual elements in the input image. This operation is extensively utilized across numerous engineering fields and serves as the foundation for computer vision algorithms.

### 1.2.1. Convolution operator

Discrete convolution represents an operation conducted between two distinct entities, usually an input and a filter, often termed as a kernel. The kernel is required to have the same dimensional count as the input. Although our discussion focuses on 2-dimensional objects, this notion can be generalized to encompass any number of dimensions. The fundamental concept involves employing a filter  $K$  and moving it across the input image  $I$  in such a way that it covers all potential positions. At every position, a multiplication operation occurs between all the elements of the kernel and the overlapping input image values, with the subsequent results being summed. This aggregated sum of products yields the output image value located at the central position of the prevailing kernel location. The associated mathematical expression can be depicted as follows:

$$(I * K)_{u,v} = \sum_{i=-h_1}^{h_1} \sum_{j=-h_2}^{h_2} I(u+i, v+j) * K(i, j) \quad (1.1)$$

where the  $K$  kernel is the following :

$$K = \begin{pmatrix} k_{-h_1, -h_2} & \cdots & k_{-h_1, h_2} \\ \vdots & \ddots & \vdots \\ k_{h_1, -h_2} & \cdots & k_{h_1, h_2} \end{pmatrix} \quad (1.2)$$

In contrast to FCNs, where weights of the affine linear transformation are learned, CNN learn and optimize filter values to extract the most salient features from input data. Multiple filters are learned within each convolutional layer, with each filter producing a single output. It is crucial to recognize that the convolution operation imparts several vital characteristics to the network:

1. Weight sharing: the number of parameters that must be defined is considerably lower compared to FCNs. As the kernels slide across the image, all pixels share the same weights, determined by the filter values. Consequently, the quantity of parameters relies solely on the filter count and dimensions. This significantly reduces computational complexity, rendering the operation more efficient and expedient.

2. Spatial invariance: the network preserves spatial consistency, processing structures independently of their position within the image. Thus, target objects are identified regardless of their location in the image.
3. Local connectivity: CNNs capitalize on the local spatial coherence properties of images and establish spatial connectivity patterns between neurons in neighboring layers. The value of each neuron in a given layer is dependent on only a few neurons in the preceding layer. In contrast to being fully connected, neurons are linked exclusively to those in the subsequent layer that are spatially proximate.

Convolutional layers carry out the convolution of inputs with a specified number of filters, yielding a number of outputs equivalent to the total convolved filters. These outputs are commonly referred to as feature maps.

The parameters of each filter are learned to extract distinct information components embedded within the input image. Initial layers predominantly concentrate on extracting low-level features, such as edges and blobs. In contrast, deeper layers can learn more abstract and high-level features, including colors, shapes, and characteristic patterns, culminating in the formulation of output segmentation masks. This process is also illustrated in Figure 1.2.

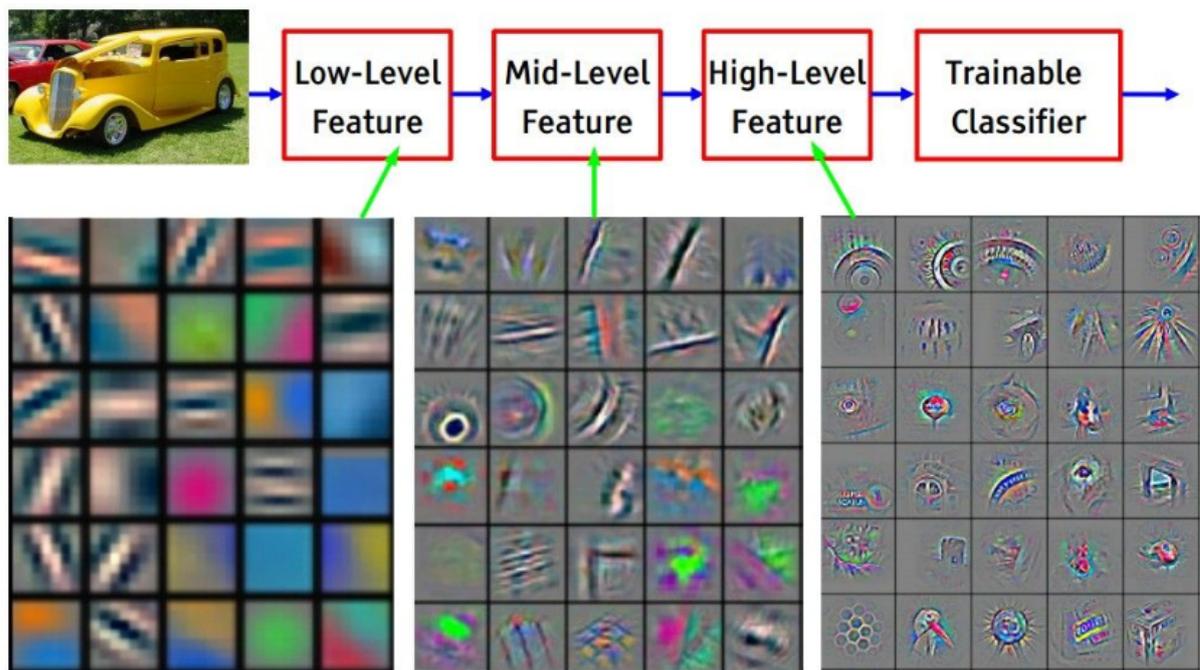


Figure 1.1: Visualization of the features from different layers of a CNN. In CNNs, the lower layers typically detect simpler features such as edges and corners, while the higher layers learn more complex features such as shapes and textures. Different filters or kernels in a layer may detect edges of different shapes, orientations, and sizes, depending on their weights and parameters. For example, a filter with a vertical orientation may detect vertical edges, while a filter with a diagonal orientation may detect diagonal edges

### 1.2.2. Pooling layer

Following each activation function, a pooling layer is incorporated to simplify the overall computational complexity of the network. The pooling operation filters and selects feature maps while downsampling the data. This procedure reduces the image resolution but expands the field of view for subsequent layers, enabling CNNs to integrate more contextual information, retain significant features, and discard less pertinent ones. In fact, the gradual diminishment of image size allows filters in deeper layers to concentrate on larger receptive fields and capture global context. Additionally, it aids in mitigating overfitting to training data. The alternating arrangement of convolutional and pooling layers progressively reduces the input size.

Various operations can be employed for pooling. Max-Pooling represents one of the most prominent pooling techniques, which involves extracting patches from input feature maps and outputting the maximum value for each channel. Alternative downsampling methods

include Average-Pooling and Global Average-Pooling. Nevertheless, max-pooling tends to deliver superior performance, as it is more beneficial for neural networks to learn the maximal presence of distinct features rather than their average presence [5].

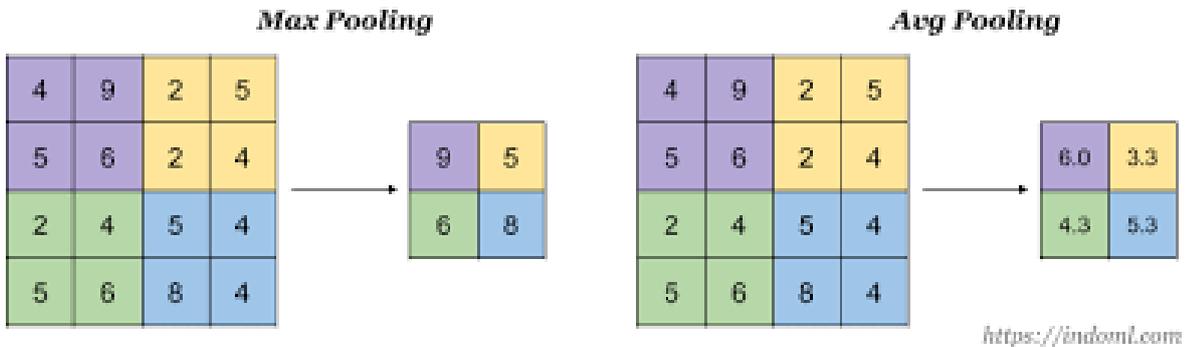


Figure 1.2: Different pooling layers. Max and average pooling.

### 1.2.3. Non linear activation function

Another important component of CNN architectures are activation functions. They are applied to the outputs of convolutional layers, mapping values depending on the morphology of the function and basically deciding whether a neuron will fire or not. These functions serve for introducing non-linearities to neurons, in order to better encode the complex representation from the input to the output of the model and recognize and learn patterns in data. Also, the generalization ability of the networks benefits from this. Some different types of activations can be used, depending on the task and the architecture choice. Bounded activations squeeze input values into an output range between 0 and 1 for Sigmoid, for example. On the other hand, unbounded functions can map inputs up to infinite. Although some smooth non linear functions were mathematically inspired by the biological neuron behavior, most of current methods are using the Rectified linear unit (ReLU) function. Authors in [6] and [7] argue that it achieves better results in deep networks as it makes the activation sparse and more efficient. This activation maps negative inputs to zero and positive inputs with an identity transformation, bringing to the following mathematical formulation, and graphic representation:

$$y = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \quad (1.3)$$

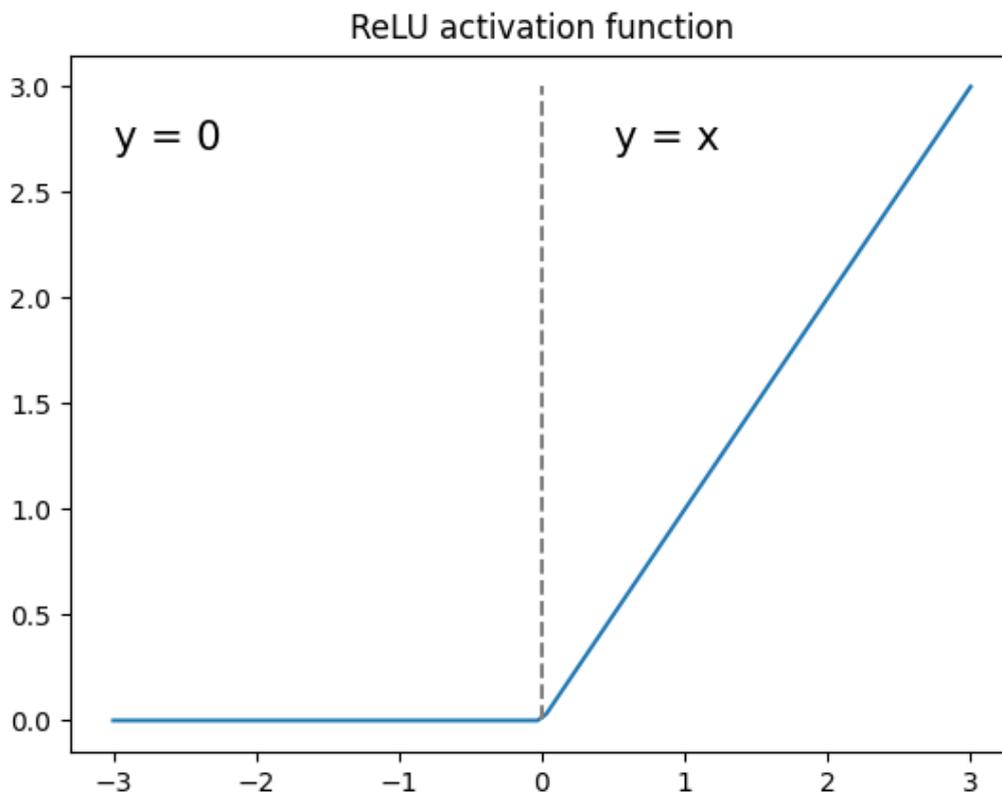


Figure 1.3: ReLu function plot in 2d axis.

#### 1.2.4. Learning rate

To minimize the network's loss function, the learning rate is employed to scale the magnitude of weight updates; it is essentially defined as the step size of the parameter updating process. The learning rate dictates the extent to which model weights are modified in response to the estimated error at each iteration. It is regarded as a critical hyperparameter to configure when training a neural network, and selecting its value can be quite challenging.

If the learning rate is set too low, training progresses at a slow pace, leading to minuscule weight updates and sluggish convergence. Conversely, if the learning rate is set too high, it may result in undesirable divergent behavior in the loss function due to various factors, such as increased sensitivity to outlier values, heightened noise in the data, and overly rapid training that prevents the network from learning generalization. When set too high, the loss function initially decreases but subsequently increases after a certain number of epochs, indicating that the network has ceased learning [8].

### 1.2.5. Optimization

The manner in which the network modifies and adjusts its parameters is governed by the gradients of the cost function. The gradient obtained through the backpropagation process is utilized by an optimization algorithm to facilitate learning and update neuron weights in order to achieve a global minimum of the loss function. Gradient Descent is one of the most widely used optimization algorithms, which relies on the first-order derivative of a loss function. It computes the necessary weight alterations to enable the function to reach its minimum value. Through backpropagation, the loss is transmitted from one layer to another, and the model's parameters are adjusted based on the losses to minimize the loss function.

Stochastic Gradient Descent (SGD) represents a variant of Gradient Descent that aims to update model parameters more frequently. In SGD, model parameters are modified following the computation of loss on individual training examples, as opposed to calculating the gradient of the cost function with respect to the parameters for the entire training set. SGD outperforms traditional gradient descent because it eliminates redundancy by conducting one update at a time. However, since model parameters are updated frequently, weights exhibit higher variance and fluctuations in loss functions, leading to a noisier system than conventional Gradient Descent. Moreover, SGD typically requires a greater number of iterations to reach the minimum [9].

## 1.3. CT scan images

Computed tomography (CT) imaging is a medical imaging technique that captures the internal anatomy of the body utilizing a combination of X-ray technology and computational processing, conducted by radiographers in clinical environments. The various anatomical structures that can be visualized by CT scans encompass internal organs, vasculature, and osseous structures, facilitating their application in diagnostic procedures, such as detecting bone fractures, organ injuries, vascular abnormalities, and neoplasms.

CT amalgamates multiple X-ray projections acquired from various angles as the X-ray tube revolves around the body, generating cross-sectional images of internal body regions. Concurrently, a detector situated opposite the X-ray tube captures the transmitted projection data. As the X-ray beam penetrates the tissue, it experiences attenuation, the extent of which depends on the tissue's density, allowing for the measurement of an attenuation coefficient. Within a CT slice, the grayscale values correspond to X-ray attenuation, representing the fraction of X-rays scattered or absorbed while traversing each voxel. The

implementation of techniques such as cone-beam, extreme multi-detector, dual-energy, and iterative reconstruction algorithms has led to significant advancements in computed tomography technology [10].

Additionally, CT scans can serve as ancillary tools for other examinations, such as monitoring neoplasm behavior following surgical intervention or ascertaining tumor dimensions, morphology, and position prior to radiotherapy. In general, CT scans are not employed for screening purposes due to their inherent risks and the potential to induce unwarranted patient stress and anxiety.

A particular category of CT imaging, as you can see in Figure 1.4, is the contrast-enhanced CT scan. Contrast-enhanced CTs enable the visualization of body structures that may be indiscernible in non-contrast-enhanced CT scans by employing substances referred to as radiocontrast agents.

## 1.4. Image segmentation

Image segmentation is the process of identifying distinct structures within an image and demarcating their boundaries to distinguish them from each other and the background. Segmentation is a classification task in which models concentrate on classifying individual pixels in an image or voxels in a volume. Each voxel is assigned a label in a process known as semantic segmentation, resulting in an image where voxels sharing the same label exhibit specific characteristics. Image segmentation offers a more meaningful data representation and is a critical step in comprehending medical image content and conducting diagnoses. Following segmentation, all separate regions should exhibit homogeneity regarding certain attributes and demonstrate spatial compactness. Effective image segmentation should fulfill several fundamental requirements:

- Every pixel in an image must belong to a class
- Each region must be homogeneous concerning specific attributes

Segmentation continues to be a widely researched topic in medical image literature. Various segmentation methods can address different challenges, and the most appropriate method should be chosen based on the specific objective. Some traditional, successful approaches will be discussed in the following sections.

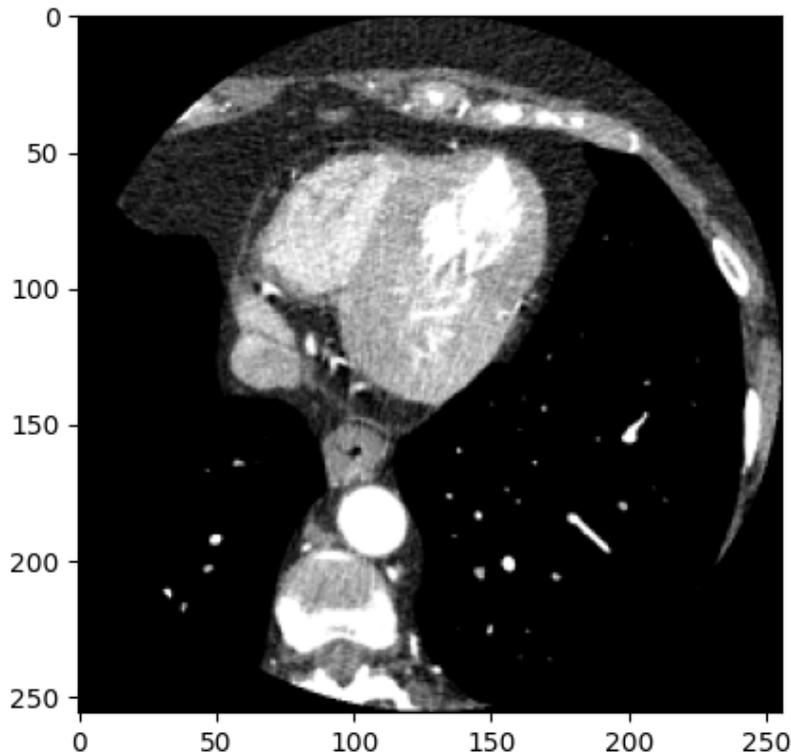


Figure 1.4: Example of a Ct scan slice from a patient.

### 1.4.1. UNet

Building on the concept of FCN deconvolution to restore image size and features, Ronneberger et al. [11] developed the U-Net in 2015, achieving impressive segmentation results without requiring a large volume of training data. The goal was to enable automatic segmentation of medical images using a CNN that produces a high-resolution segmentation mask as output. U-Net employs an encoder-decoder structure similar to a simple FCN, consisting of two roughly symmetric paths—descending and ascending. The encoder progressively reduces the spatial dimensions by continuously merging layers to extract feature information, a process known as downsampling. The decoder, referred to as upsampling, incrementally restores target details and spatial resolution.

The original U-Net architecture is depicted in Figure 1.5. The encoding path adheres to the standard CNN architecture, featuring five consecutive depth levels. In the descending path, the network performs two  $3 \times 3$  convolutions at each step, doubling the number of filters to expand the feature space extracted from the data. Each convolution operation

is followed by a ReLU activation function, and a  $2 \times 2$  max-pooling operation with strides 2 is added, reducing the feature map size by half. At the end of the contracting path, the image is resized from  $512 \times 512 \times 1$  to  $32 \times 32 \times 1024$ . In the expansive path, the image is resized to its original dimensions through the application of transposed convolutions, which enlarge the spatial representation of data and incrementally increase image size.

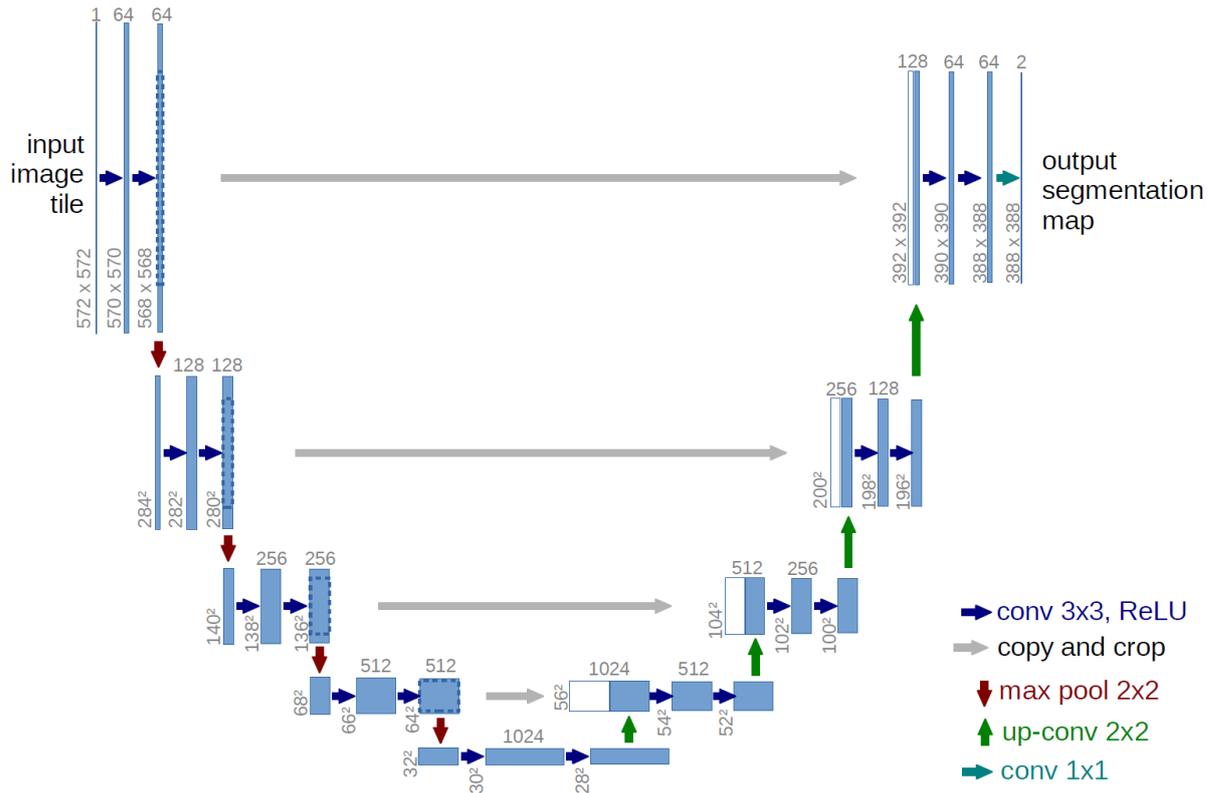


Figure 1.5: UNet original architecture.

After transposed convolution, the image is up-sized and concatenated with the corresponding image from the contracting path. Concatenation, a form of skip connection, enables subsequent layers to reuse intermediate representations learned from earlier layers, leading to improved model performance and better gradient propagation across the network. This type of skip connection is widely used in DenseNets and Inception networks [12]. In the U-Net, skip connections combine information from previous layers to generate more accurate predictions. Concatenation allows for localized information, making semantic segmentation possible. Encoder layers are skip-connected and concatenated with decoder layers, combining spatial information from the down-sampling and up-sampling paths to retain optimal spatial information. This direct connection between opposing contracting and expanding convolutional layers enhances the U-Net's segmentation accuracy [13]. Following concatenation, the feature map is deconvolved, and the final expanding layer

performs a 1x1 convolution to produce the segmentation map.

In the original paper, the number of training samples was limited. The authors addressed this issue by applying random elastic deformation to the training data, enabling the network to learn variations without needing additional labeled data. Furthermore, they implemented a weighted loss function that penalizes the model for failing to separate two objects, as separating touching objects of the same class is a significant challenge in semantic segmentation [14].

## 1.5. RNN

Recurrent neural networks (RNNs) are a family of networks designed for processing sequential data, which refers to a series of values  $x(1), x(2), \dots, x(\tau)$  generated by a temporal process. Like CNNs, RNNs employ parameter sharing, but instead of processing different input portions, the sharing occurs across various time steps of a sequence. This is achieved by making each output a function of the input at time index  $t$ , the output or outputs at a previous time step  $t' < t - 1$ , and using the same function to compute the output at each sequence index. Consequently, RNNs implement a recurrent formulation, resulting in parameter sharing through a deep computational graph.

Various RNN architectural variations exist, but most share the idea of computing the output for the next sequence element, considering the entire sequence history seen by the network. This is accomplished using a special variable called the network's state, which is essentially the value of the intermediate layers' outputs (hidden units). The state/value of hidden units  $h(t)$  at time index  $t$  can be expressed as a vector

$$h(t) = g(t)(x(t), x(t-1), x(t-2), \dots, x(2), x(1)) = f(h(t-1), x(t), \theta) \quad (1.4)$$

RNNs use the state as a lossy summary of the input/output history, as sequences of any length are mapped to a fixed-length variable, the hidden state  $h(t)$ . This mapping allows the network to generalize information from sequences of varying lengths, as a single model  $f$  operates on all indexes instead of numerous functions  $g(t)$  for each time step  $t$ .

Different design patterns for RNNs exist, including connections between hidden units, output recurrence, and input-output sequence length mismatches.

An essential operation in using RNNs is unfolding. This entails repeating the graph across different time steps, which is necessary for numerical implementation of the graph. Unfolding enables training through an adapted version of the backpropagation algorithm

called backpropagation through time (BPTT). BPTT is not a specialized algorithm but rather an application of backpropagation in the unfolded RNN computational graph, considering each time step of the sequence.

### 1.5.1. LSTM

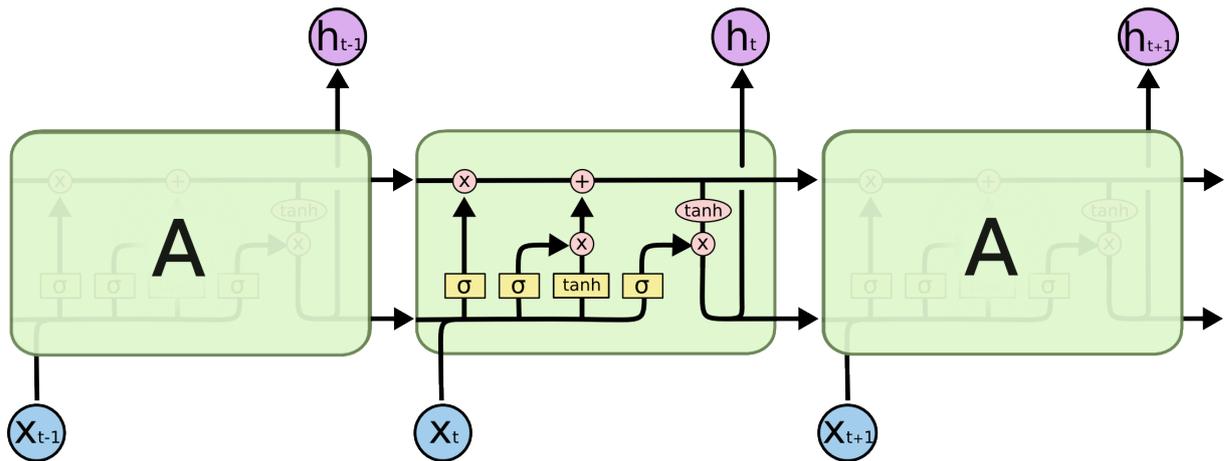


Figure 1.6: LSTM architecture

The Long Short-Term Memory (LSTM) model, initially introduced in 1997 by Hochreiter et al. [15], has been extensively employed and has demonstrated remarkable performance in various tasks involving sequence data [16]. The LSTM architecture incorporates three distinct gating mechanisms, which can be perceived as layers since they consist of a linear transformation, followed by an activation function (specifically, a sigmoid in this case), and finally, a point-wise element multiplication. These gates control separate aspects of the variables utilized by the function executed by a recurrent layer, and they are specifically referred to as:

- Forget gate: responsible for determining which elements of the previous state  $h(t-1)$  should be retained or discarded;
- Input gate: focuses on determining the relevance of elements from the new input  $x(t)$  for the task at hand.
- Output gate: ultimately decides the significance of the elements in the new output  $y(t)$ .

The full explanation on the gates can be found in [17].

### 1.5.2. Motivation of the study

Cardiovascular disease is the leading cause of mortality worldwide, and early detection and accurate diagnosis are essential for successful treatment and patient management. Cardiac CT imaging is a non-invasive and widely used imaging modality for the evaluation of cardiovascular disease. One important task in cardiac CT imaging is the segmentation of the left ventricle, which provides valuable information about cardiac function and morphology. Manual segmentation of the left ventricle is time-consuming and subject to inter- and intra-observer variability. Automated segmentation using deep learning models has the potential to improve the efficiency and accuracy of this task.

The main motivation for this study is to investigate the potential of deep learning models for the segmentation of the left ventricle in cardiac CT images. Specifically, this thesis aims to compare the performance of three deep learning models for this task: U-Net, U-Net 2.5D, and U-Net with LSTM. The study also aims to provide insights into the strengths and weaknesses of each model and to evaluate their suitability for clinical applications.

The research questions addressed in this study are:

- The comparison of the models (U-Net, U-Net 2.5D, and U-Net with LSTM) in terms of their performance for the segmentation of the left ventricle in cardiac CT images.
- The strengths and weaknesses of each model in terms of accuracy, efficiency, and generalization.

The answers to these research questions will contribute to the development of more accurate and efficient methods for the segmentation of the left ventricle in cardiac CT images, which can improve patient outcomes and aid in the diagnosis and management of cardiovascular disease.

## 2 | Materials and Methods

### 2.1. Population

The training, validation, and test sets were created using the dataset utilized in this study. Although each patient has many photos or slices, they were separated based on the patient rather than across sets. Overfitting would have happened if the dataset had been split into random images or slices instead of people. In fact, using the same patient to train and test the model would cause a data leak that wouldn't be representative of the population as a whole and would give an inaccurate picture of how well the model worked.

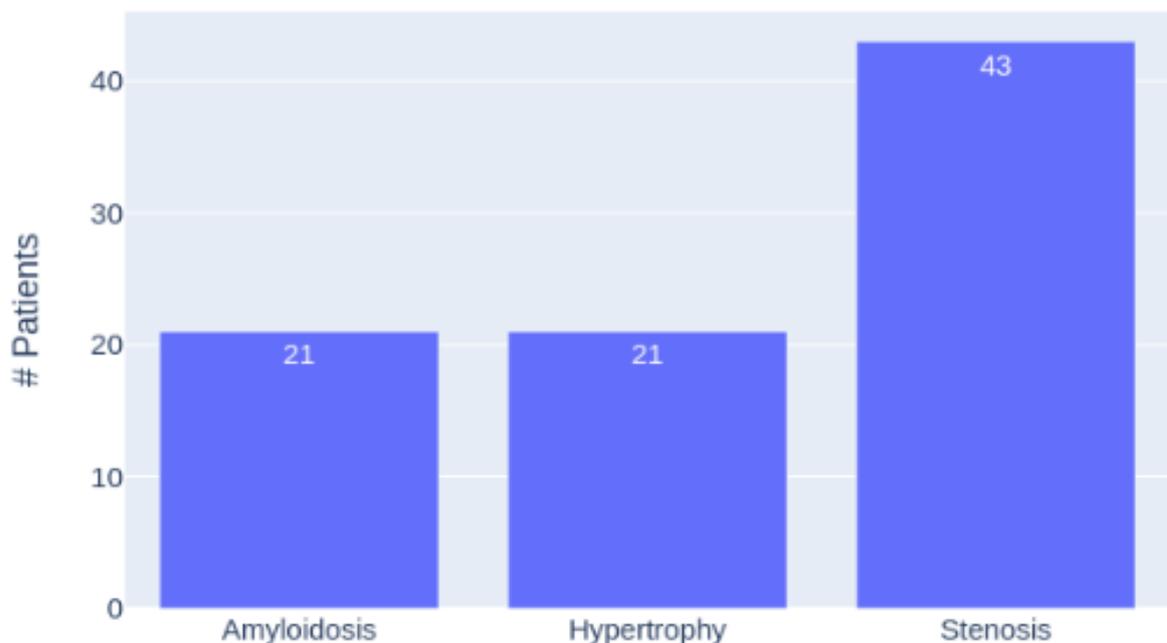


Figure 2.1: Barplot that represents the number of patients for each pathology.

The dataset contains a total of 85 patients, spread across three pathologies: amyloidosis, hypertrophy, and stenosis. The amyloidosis and hypertrophy categories each have 21 people, whereas the stenosis category has 43 patients as illustrated in Figure 2.1. The number of patients for each set was determined as follows: 75% for training, 15% for

validation, and 10% for testing. The dataset was then randomly shuffled. . There are 6171 photos in total, with 72.2 slices on average per subject as shown in Figure2.3. The 3D CT scan volume is made up of a variable number of 2D images for each patient. According to the distribution of images and slices by category, amyloidosis accounts for 26.1%, hypertrophy for 22.8%, and stenosis for 51.1%. This result shows that the stenosis category covers the majority of the dataset.

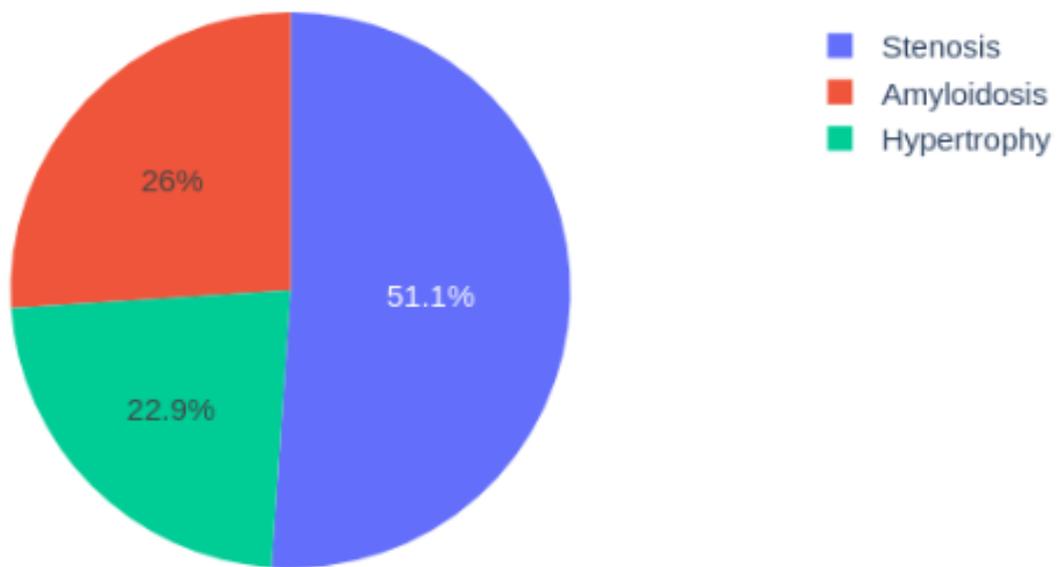


Figure 2.2: Pie chart that represents the percentage of images per each pathology

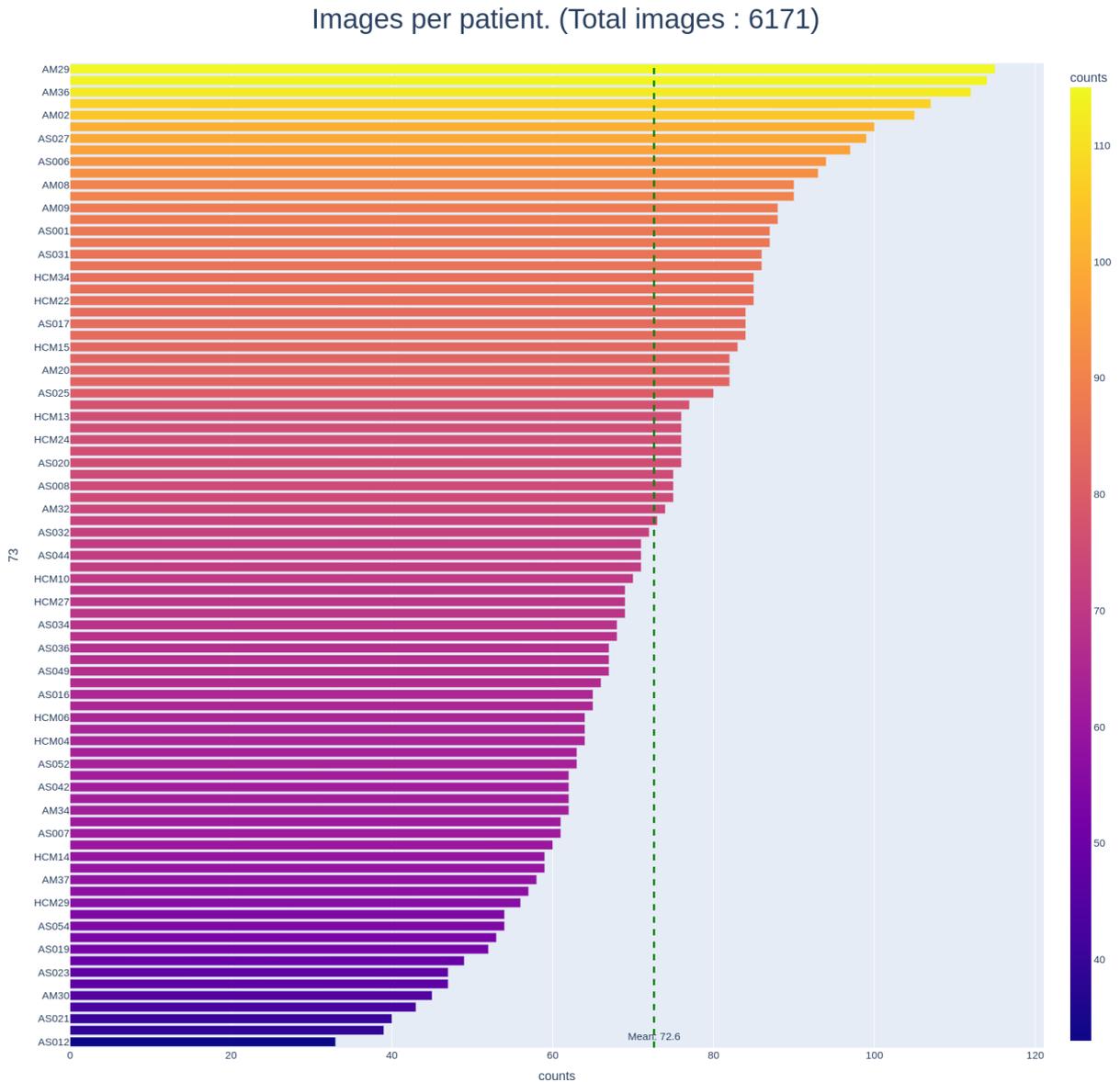


Figure 2.3: Number of slices having the manual target segmentation per patient

## 2.2. CT scan protocol

The dataset used in this study consisted of cardiac CT images (CCT) acquired from two different CT scanners: a 256-slice scanner (Revolution CT; GE Healthcare, Milwaukee, WI) and a 320-slice wide volume coverage scanner (Aquilion ONE Vision™; Canon Medical Systems Corp., Tokyo, Japan). No premedication with beta-blockers or nitrates was administered before CT acquisition.

The Revolution CT scans were obtained with the following acquisition parameters: peak

tube voltage of 100 kV; detector collimation of 160 mm using 256 rows by 0.625 mm on the Z-axis; detector geometry of 256 rows by 832 detection elements per row; high contrast spatial resolution of 0.23 mm; slice thickness of 0.625 mm; gantry rotation time of 280 ms; prospective triggering; and iterative reconstruction algorithm (ASIR-V; GE Healthcare). Patients received a fixed dose of 50 ml bolus of contrast medium (400 mg of iodine per milliliter, Iomeprol; Bracco, Milan, Italy) regardless of BMI via an antecubital vein at an infusion rate of 5 ml/s followed by 50 ml of saline solution at 5 ml/s.

The Aquilion ONE Vision scans were acquired with the following acquisition parameters: peak tube voltage of 120 kV; tube current-time product of 160 mAs; section collimation of 320 detector rows and 1.2-mm section thickness; gantry rotation time of 275 ms

## 2.3.

## 2.4. Image Preprocessing

This section describes all the preprocessing transformations performed on our input data. The exploratory analyses and data visualizations are conducted using Dash Plotly [18], a popular library renowned for creating visually appealing dashboards.

### 2.4.1. CT Slice Order and Dataset Cleaning

In medical imaging, the sequence of CT slices is a crucial component of picture processing. The effectiveness of machine learning models is impacted by the precise slice order, which is also important for the diagnosis of the medical condition. The Slice Location attribute of the DICOM header controls the slice order [19]. The slice's location in relation to the coordinate system's origin is specified by the attribute Slice Location. It is determined as the distance along the normal vector to the image plane from the origin to the image's center. Based on their Slice Location values, the slices are organized in either ascending or descending order.

The Slice Location element of the DICOM header is used in this project to establish the slice order. The Pydicom library is used to read DICOM files [20], and each slice's Slice Location property is extracted. After sorting the slices according to their Slice Location values, the 3D volume is recreated using the resultant list.

Lastly, various artifacts are eliminated from the photos to guarantee the dataset's quality. These isolated, spots that are not part of the primary image are called artifacts, and they can be brought on by noise or other elements. By applying morphological processes like

erosion and dilation, which help to remove small, isolated portions from the images while maintaining the larger sections, these artifacts are eliminated [21]. In image processing and computer vision, morphological operations such as erosion and dilation are commonly used to remove small, isolated regions (or "artifacts") from images while preserving the larger connected regions.

Erosion is a morphological operation that erodes the boundaries of foreground (non-zero) regions in an image by removing pixels near the boundaries. This can be achieved by applying a structuring element (a small binary image) to each pixel in the image and setting the pixel to zero if the structuring element overlaps with any background (zero-valued) pixels. Erosion can be useful for removing small isolated regions or thin protrusions from an image.

Dilation, on the other hand, is a morphological operation that "dilates" the boundaries of foreground regions by adding pixels near the boundaries. This can be achieved by applying a structuring element to each pixel and setting the pixel to one if any part of the structuring element overlaps with any foreground pixels. Dilation can be useful for filling in small gaps or holes in foreground regions.

By applying these morphological operations to an image, small, isolated portions of the image (such as noise or artifacts) can be removed while maintaining the larger connected regions. This is because the erosion and dilation operations tend to "smooth out" the boundaries of foreground regions and fill in gaps, making the connected regions more robust to noise and other artifacts.

Overall, crucial steps in getting the dataset ready for deep learning models include putting the slices in the right sequence, removing empty masks, and cleaning up the images of artifacts. These procedures guarantee a high-quality dataset and efficient model learning from the data.

### 2.4.2. Hounsfield unit transformation (HU)

HU is a unit of measurement used in CT scans to quantify the density of various tissues based on the attenuation of X-rays. It is named after the British physicist Sir Godfrey Hounsfield who developed the CT scanner. The HU scale is defined such that the density of water at room temperature is 0 HU, while the density of air is -1000 HU, and that of bone is around +1000 HU.

The CT scanner measures the amount of X-ray attenuation at different points in the body and assigns a HU value to each point based on the difference between its attenuation and

that of water. The raw CT images acquired from the scanner typically have pixel values that are proportional to the HU scale. However, the actual HU values of the tissues can vary depending on the calibration of the scanner and the patient's anatomy. Therefore, the HU values need to be corrected by applying a linear transformation that maps the raw pixel values to the actual HU values.

The HU transformation involves multiplying the raw pixel values by the rescale slope and adding the rescale intercept value. The rescale slope and intercept values are stored in the DICOM header of the CT image and are used to calibrate the pixel values to the HU scale. The HU transformation ensures that the HU values of the tissues in the image are consistent across different CT scanners and patients.

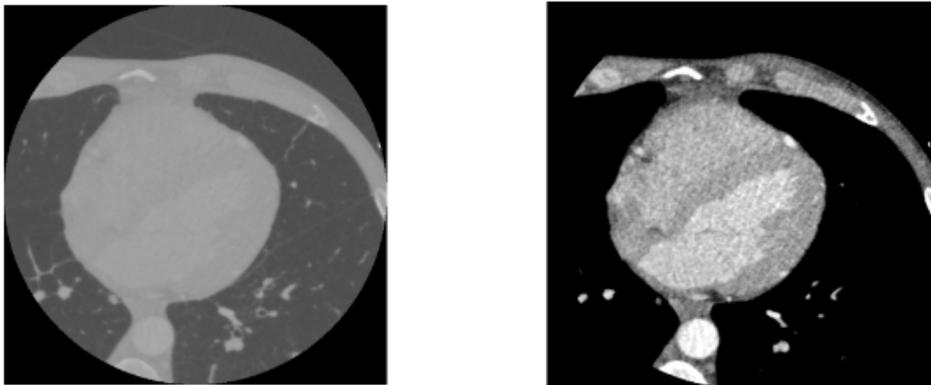


Figure 2.4: Comparison of the preprocessing. On the left we can see the original image in which is hard to distinguish the ventricle. On the right instead is much easier to distinguish it

### 2.4.3. Windowing

Windowing is a technique used to adjust the contrast and brightness of medical images, such as CT and MRI scans. It involves mapping the original pixel values to a new range of values based on the desired contrast and brightness levels. The window level represents the center of the new range, while the window width represents the width of the range.

In the context of CT images, the window level and width are typically chosen to highlight specific tissues or structures of interest. For example, to visualize lung tissue, a window level of -600 HU and a window width of 1500 HU may be used. This maps the pixel values corresponding to the air in the lungs to the lower end of the new range, and the

pixel values corresponding to the surrounding tissue to the higher end, resulting in a high contrast image of the lungs.

The window level and width values of 40 and 400 respectively are commonly used for CT scans of the heart and ventricles, as they provide good contrast between the blood pool and the surrounding myocardium and other tissues [22].

**Window level:** the window level represents the midpoint of the range of pixel values that will be displayed on the image. A window level of 40 HU means that the midpoint of the displayed range is set to 40 HU, which corresponds to the HU value of blood in CT scans. This means that the blood pool will be displayed with a higher brightness level compared to the surrounding tissues, which enhances its visibility and makes it easier to identify the heart chambers.

- **Window level:** the window level represents the midpoint of the range of pixel values that will be displayed on the image. A window level of 40 HU means that the midpoint of the displayed range is set to 40 HU, which corresponds to the HU value of blood in CT scans. This means that the blood pool will be displayed with a higher brightness level compared to the surrounding tissues, which enhances its visibility and makes it easier to identify the heart chambers.
- **Window width:** the window width represents the width of the range of pixel values that will be displayed on the image. A window width of 400 HU means that the displayed range extends from 200 HU below the window level (i.e.,  $40 - 200 = -160$  HU) to 200 HU above the window level (i.e.,  $40 + 200 = 240$  HU). This width is wide enough to include most of the pixel values corresponding to the blood pool and myocardium, while excluding the pixel values corresponding to other tissues such as lungs and bones. This results in a high contrast image of the heart and ventricles, which makes it easier to visualize and analyze their structure and function. Nonetheless, the values of 40 and 400 have proven suitable for numerous applications and commonly serve as a preliminary point for further adjustments and optimization [23].

The comparison is shown in Figure 2.5.

#### 2.4.4. Data Augmentation Techniques

In computer vision, data augmentation is a popular technique that involves creating new training images by applying various transformations to the original images. By using data augmentation, the size of the training dataset can be artificially increased, which can

improve the accuracy and generalization performance of machine learning models. This technique can enhance the performance of deep learning models by providing more diverse and representative examples of the underlying data distribution. The Albumentations library is a popular Python library for data augmentation [24].

This library has a number of image enhancements that can be applied to training data through a pipeline. The pipeline is made up of a series of transformations that are sequentially applied to each image in the dataset. The transformations used in this project are as follows, applied in the specified order:

- **Resize:** the image is resized to a square with a resolution of 256x256 pixels.
- **Horizontal Flip:** the image is flipped horizontally with a probability of 0.3. This is done to ensure that all images are the same size and to lessen the computational complexity of the model. Using this transformation, the training set's diversity is increased and overfitting is decreased.
- **Vertical Flip:** the image is flipped vertically with a probability of 0.3. This transformation is similar to the horizontal flip but applied in the opposite direction.
- **Safe Rotate:** the image is rotated at random with a probability of 0.2 at an angle between -90 and 90 degrees. This transformation is used to broaden the training set's diversity and reduce overfitting.

These transformations have been selected based on how well they increase the diversity of the training dataset and enhance the model's generalization capabilities.

## 2.5. Evaluation Metrics in Segmentation

Segmentation is a crucial task in medical image analysis, where the objective is to identify and delineate different anatomical structures or pathologies in an image. The performance of a segmentation algorithm is evaluated using various metrics, which quantify the accuracy and robustness of the algorithm.

### Pixel Precision and Recall

Pixel precision and recall are common metrics used in segmentation to evaluate the accuracy of the segmentation results at the pixel level.

Pixel precision is defined as the ratio of true positive pixels to the total number of pixels predicted as positive:

$$Precision = \frac{TP}{TP + FP} \quad (2.1)$$

where TP is the number of true positive pixels and FP is the number of false positive pixels. Pixel recall is defined as the ratio of true positive pixels to the total number of positive pixels in the ground truth:

$$Recall = \frac{TP}{TP + FN} \quad (2.2)$$

where FN is the number of false negative pixels. These metrics provide a measure of the algorithm's ability to accurately detect and classify each pixel in the image.

## F1 Score

The F1 score is a metric that combines precision and recall to evaluate the overall performance of a segmentation algorithm. It is defined as the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (2.3)$$

The F1 score is a commonly used metric in segmentation, as it provides a balanced measure of the algorithm's performance.

## Intersection over Union (IoU)

The intersection over union (IoU) metric is another popular metric used in segmentation, which measures the overlap between the predicted and ground truth segmentation masks. It is defined as the ratio of the intersection of the two masks to their union:

$$IoU = \frac{TP}{TP + FP + FN} \quad (2.4)$$

The IoU metric provides a measure of the algorithm's ability to accurately segment the object of interest, while accounting for variations in size and shape of the object.

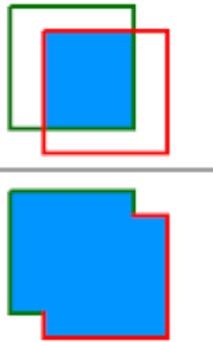
$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{Diagram 1}}{\text{Diagram 2}}$$


Figure 2.5: Visual representation of the IOU. Overlap over union.

## Dice Coefficient

The Dice coefficient is another metric used in segmentation, which measures the similarity between the predicted and ground truth segmentation masks as shown in Figure 2.7. It is defined as twice the intersection of the two masks divided by their sum:

$$Dice = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (2.5)$$

The Dice coefficient is a commonly used metric in medical image segmentation, as it provides a robust measure of the algorithm's performance.

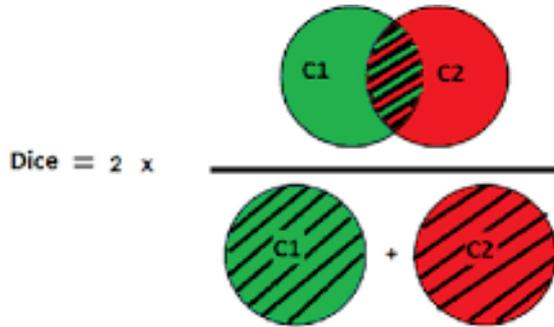
$$Dice = 2 \times \frac{\text{Diagram 1}}{\text{Diagram 2}}$$


Figure 2.6: Visual representation of the Dice coefficient.

### 2.5.1. Loss Functions for Image Segmentation

#### Binary Cross Entropy Loss

Binary Cross Entropy (BCE) is a widely used loss function for binary image segmentation [25, 26]. It is computed by taking the negative log likelihood of the predicted probability

of each pixel belonging to a certain class, summed over all pixels in the image. The BCE loss function is defined as:

$$\text{BCE}(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (2.6)$$

where  $y_i$  is the ground truth label of the  $i$ -th pixel and  $\hat{y}_i$  is the predicted probability of the  $i$ -th pixel belonging to the foreground class, with  $N$  being the total number of pixels in the image.

## Soft Dice Loss

Soft Dice Loss is another popular loss function for image segmentation [25, 26]. It is based on the Dice coefficient, which calculates the amount of overlap between segmentations predicted and observed. The Soft Dice Loss is calculated by averaging the Dice coefficients over all of the pixels in the image. The Soft Dice Loss function is defined as:

$$\text{SoftDice}(y, \hat{y}) = 1 - \frac{2 \sum_{i=1}^N y_i \hat{y}_i + \epsilon}{\sum_{i=1}^N y_i + \sum_{i=1}^N \hat{y}_i + \epsilon} \quad (2.7)$$

where  $y_i$  is the ground truth label of the  $i$ -th pixel and  $\hat{y}_i$  is the predicted probability of the  $i$ -th pixel belonging to the foreground class, with  $\epsilon$  being a small constant to avoid division by zero.

Both Binary Cross Entropy and Soft Dice Loss have their advantages and disadvantages depending on the specific task and dataset. Binary Cross Entropy is easy to compute and is commonly used as a baseline loss function for binary image segmentation. Soft Dice Loss, on the other hand, can handle class imbalance and is more robust to noisy labels.

Overall, the choice of loss function should be based on the specific requirements of the image segmentation task and the characteristics of the dataset.

## 2.6. Models

### 2.6.1. UNet

In this section, we will discuss the architecture and regularization used in the Baseline-UNet model. The UNet architecture proposed by Ronneberger et al. [25] is a popular choice for image segmentation tasks. It is a fully convolutional neural network that

consists of a contracting path and an expansive path. The contracting path is used to capture the context of the input image, while the expansive path is used to generate the segmentation mask.

The Baseline-UNet model used in this study is a variant of the original UNet architecture, with some modifications. The model takes a single-channel image as input and outputs a segmentation mask with the same dimensions as the input image. The contracting path of the model consists of four blocks, each consisting of two convolutional layers followed by a ReLU activation function and batch normalization. Each block is followed by a max-pooling layer that downsamples the feature maps. The expansive path consists of four blocks, each consisting of a transpose convolutional layer and two convolutional layers, followed by ReLU activation and batch normalization. Each block also concatenates the feature maps from the corresponding block in the contracting path. The final output of the model is obtained by a  $1 \times 1$  convolutional layer that maps the feature maps to the number of output classes.

Regularization techniques are used to prevent overfitting in the model. Dropout [27] is used after each convolutional layer in the contracting path with a probability of 0.5. L2 regularization is also applied to the convolutional layers with a weight decay of 0.0001. Batch normalization is also used to improve the training stability of the model and reduce the dependence on the initialization of the weights [28].

Overall, the Baseline-UNet model is a well-established architecture that has shown good performance in various image segmentation tasks. The modifications made to the original UNet architecture, such as the use of dropout, L2 regularization, and batch normalization, are common techniques used to prevent overfitting and improve the training stability of deep learning models. The data augmentation techniques used during training also help to improve the generalization performance of the model.

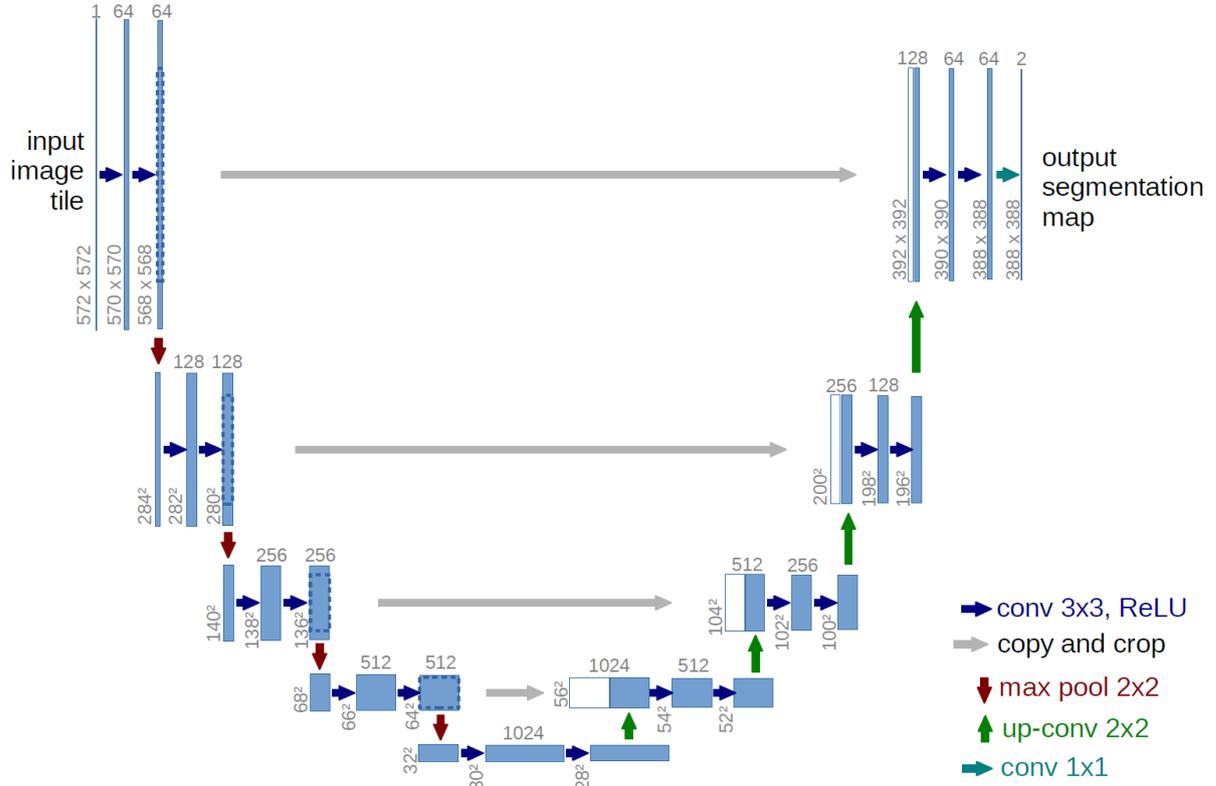


Figure 2.7: Basic UNet architecture.

### 2.6.2. UNet 2.5D

The UNet2.5D, is a variant of the UNet architecture that was designed to exploit the temporal dependency between consecutive CT slices. The idea behind the model is to use three or more consecutive CT slices as input, where the middle slice is used to predict the corresponding segmentation mask. This approach is called the 2.5D approach and has been shown to improve the segmentation performance compared to using a single slice as input [29].

The UNet 2.5D model architecture consists of an encoder and a decoder pathway, similar to the original UNet architecture [25]. The encoder pathway consists of four down-blocks, where each down-block consists of two convolutional layers, followed by a ReLU activation function and batch normalization. Each down-block also includes a max-pooling layer that downsamples the feature maps. The decoder pathway consists of three up-blocks, where each up-block consists of a transposed convolutional layer, followed by two convolutional layers, ReLU activation, and batch normalization. The up-blocks also include skip connections that concatenate the corresponding feature maps from the encoder pathway.

The main difference between the UNetGeneric model and the original UNet architecture

is that the input to the model consists of three consecutive CT slices instead of a single slice. The model takes in a 3-channel input, where each channel represents a consecutive CT slice. The model outputs a single-channel segmentation mask, which corresponds to the middle slice of the input.

The difference between the different input shape is shown in Figure 2.8 [29]

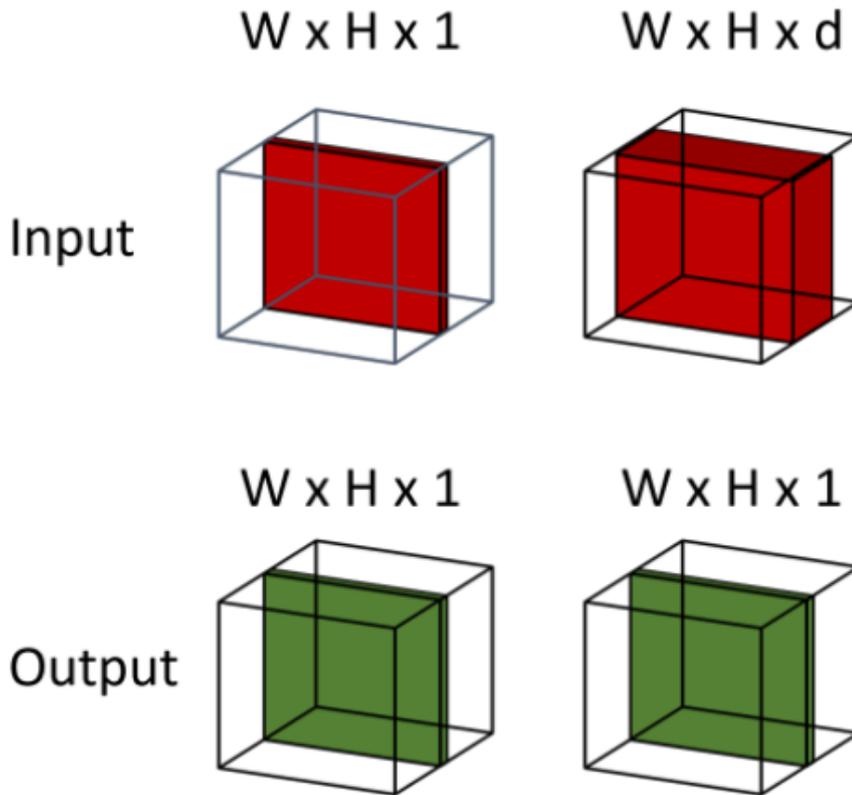


Figure 2.8: Different input/output for the UNet, UNet2.5D

### 2.6.3. UNet-LSTM

The architecture of the UNet LSTM model is a variation of the popular UNet architecture that includes LSTM (Long Short-Term Memory) layers for modeling temporal dependencies in the data [15, 30]. Here are some of the key features of this architecture:

- Input shape: the input shape is specified as  $(N, 256, 256, 1)$ , where "N" is the number of slices of the sequence. The model expects a 3D volume of shape  $(256, 256, 1)$  at each time step.
- Contraction path: the model begins with a series of convolutional layers that reduce the spatial dimensions of the input data while increasing the number of channels.

The first convolutional layer has 16 filters, and each subsequent layer doubles the number of filters until reaching 256 filters in the final layer of the contraction path. The convolutional layers are followed by batch normalization and max pooling operations to further reduce the spatial dimensions of the data.

- LSTM component: the output of the final convolutional layer in the contraction path is fed into an LSTM layer with 256 filters [15]. The LSTM layer is designed to capture temporal dependencies in the data, which can be useful in applications where the input data is a sequence of images or volumes.
- Expansive path: the model then uses a series of transpose convolutional layers to gradually increase the spatial dimensions of the data while decreasing the number of channels. The goal of this pathway is to generate a prediction for each time step in the input sequence. The transpose convolutional layers are similar to the convolutional layers in the contraction path, but they perform the opposite operation by increasing the spatial dimensions of the data. Each convolutional layer in the expansive path is followed by batch normalization.
- Output: the final output of the model is a binary mask that indicates the segmentation of the input data at each time step. The mask is generated using a single convolutional layer with one filter and a sigmoid activation function [30].

Overall, the UNet LSTM model combines the strengths of UNet's ability to generate accurate segmentations with LSTM's ability to model temporal dependencies in the data. The model is well-suited to applications where the input data consists of a sequence of images or volumes, such as in medical imaging or video analysis.

An illustration of a high level view of the model is shown in figure 2.9

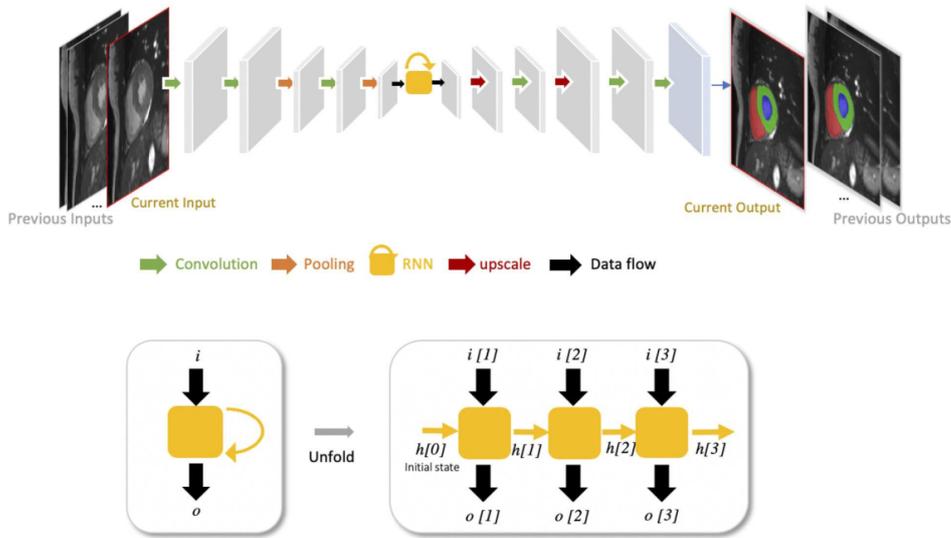


Figure 2.9: Example of UNet-lstm architecture from [31]

## 2.7. GUI for Mask Visualization

In order to facilitate the visualization and comparison of the results of the different trained models, we developed a graphical user interface (GUI) using the Dash framework in Python. The GUI allows the user to select a patient and the segmentation masks to be displayed for comparison. The user can also slide through the different slices of the CT scan to visualize the structure of the left ventricle.

It was developed using the Plotly library for creating interactive plots of the CT scan and segmentation masks, and the Dash library for the overall web layout.

The application consists of two main components: a sidebar and a content section. The sidebar allows the user to select the patient and the segmentation masks to be displayed, while the content section displays the CT scan and the corresponding segmentation masks.

It also includes a slider that allows the user to move through the different slices of the CT scan. When the slider is moved, the corresponding CT scan and segmentation masks are updated accordingly. The user can also select or deselect the different segmentation masks to be displayed using the checklist in the sidebar.

Overall, the GUI provides an easy-to-use interface for the visualization and comparison of the results of the different trained models.

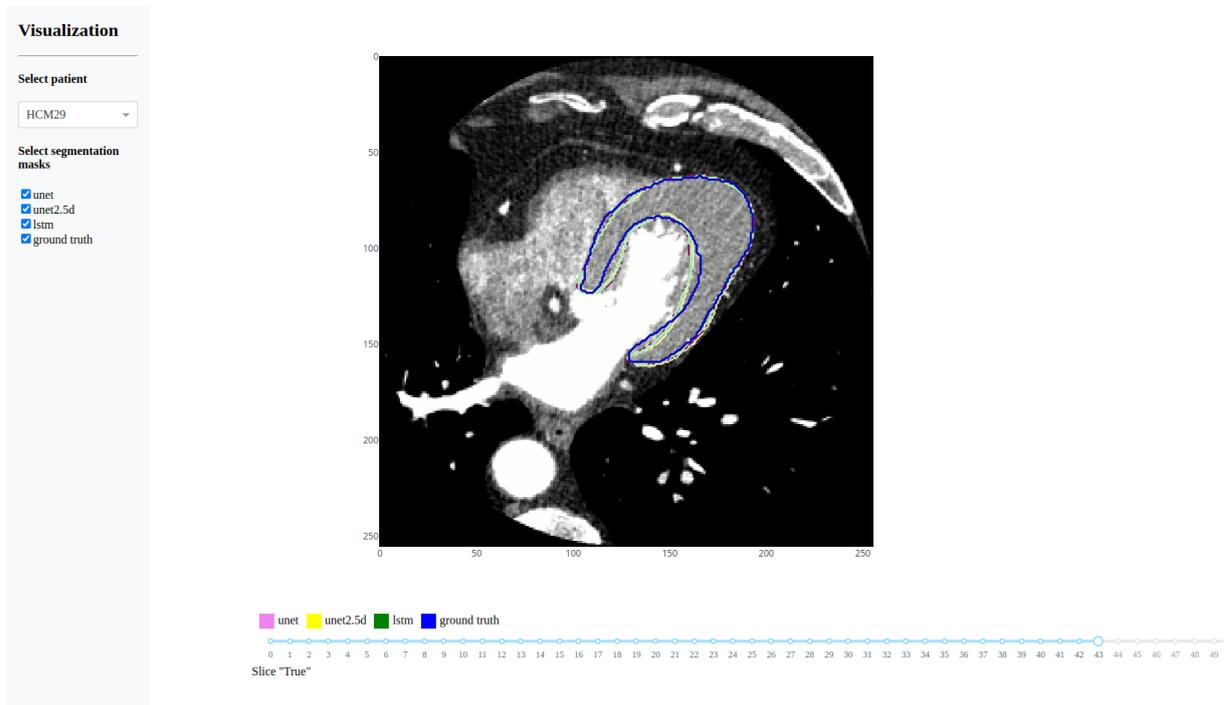


Figure 2.10: The GUI allows selecting a patient and the segmentation models' masks to display. A slider enables browsing through the slices of the selected patient.



# 3 | Results

The results shown in this study include the performance of the best models for each of architecture. We evaluated the performance of three segmentation models: UNet, UNet 2.5D, and LSTM. Specifically, these models were selected based on their performance on the validation set. The models were ranked based on their performance on the evaluation metrics, with the best model being the one that achieved the highest score on these metrics. The metrics used are the Dice Coefficient and the Intersection over union. The section provides a comprehensive evaluation of the segmentation models used in this study. Overall, the results demonstrate the effectiveness of the evaluated models and provide valuable insights for future research in this area.

## 3.1. Post processing

After generating the segmentation results by the models, post-processing steps were applied as described in the Materials and Methods section. Specifically, we applied a variation of the Largest Component Extraction Algorithm (LCEA). The LCEA is designed to process a binary mask and output a new binary mask containing only the largest connected component applied in order to obtain a single, unified component for each segmentation. These steps were necessary to ensure that the resulting segmentations accurately represented the target anatomical structures.

As an example, consider the Figure 3.1 of patient HCM27, where a small, incorrect second component was present in the segmentation. This component was identified and removed during the component analysis step, resulting in a final segmentation that accurately represented the target anatomy.



Figure 3.1: This picture represents the modification applied by the post processing. We can see that a second component is correctly deleted by the post processing.

## 3.2. Validation

### 3.2.1. UNet

Table 3.1 shows the results of the U-Net model on a per-patient basis. The table includes the number of images for each patient, as well as the mean and standard deviation of the Dice coefficient and the IoU. The results show that the U-Net model achieved varying levels of performance across the different patients. Patients HCM29 and AS043 had the highest mean Dice coefficient and IoU, while patient HCM05 had the lowest mean Dice coefficient and IoU. Overall, the U-Net model achieved a mean Dice coefficient of 0.864 and a mean IoU of 0.763 across all patients. The standard deviations of these metrics indicate that the model's performance did not vary considerably across different images within each patient, with some images achieving high scores while others achieved low scores.

Patient	N images	Dice	IoU
HCM29	56	0.895±0.053	0.814±0.080
HCM27	68	0.859±0.038	0.755±0.058
HCM05	73	0.767±0.063	0.627±0.084
HCM10	70	0.871±0.032	0.772±0.050
HCM30	47	0.883±0.031	0.792±0.049
AM08	90	0.868±0.037	0.769±0.059
AM01	39	0.852±0.034	0.743±0.051
AM37	58	0.852±0.036	0.744±0.055
AS030	60	0.896±0.030	0.812±0.050
AS031	58	0.857±0.026	0.751±0.040
AS043	67	0.899±0.026	0.818±0.041
Mean	62.364	0.864±0.037	0.763±0.056

Table 3.1: Metric results for UNet

### 3.2.2. UNet 2.5D

Table 3.2 summarizes the performance of the UNet 2.5D model in the validation set. The table shows the patient identifier, the number of images for each patient, the mean and standard deviation of the Dice coefficient, and the mean and standard deviation of the Intersection over Union (IoU) coefficient. Overall, the UNet 2.5D model achieved a mean Dice coefficient of 0.874 and a mean IoU coefficient of 0.780, with standard deviations of 0.031 and 0.048, respectively. The model performed well on most patients, with mean Dice and IoU coefficients ranging from 0.849 to 0.905 and from 0.739 to 0.828, respectively. However, some patients, such as HCM05, had lower performance, with mean Dice and IoU coefficients of 0.791 and 0.658, respectively. These results suggest that the UNet 2.5D model may be a suitable option for left ventricle segmentation.

Patient	N images	Dice	IoU
HCM29	56	0.914±0.032	0.843±0.052
HCM27	68	0.870±0.034	0.771±0.054
HCM05	73	0.791±0.053	0.658±0.075
HCM10	70	0.880±0.030	0.788±0.048
HCM30	47	0.886±0.029	0.796±0.047
AM08	90	0.879±0.027	0.786±0.043
AM01	39	0.861±0.023	0.757±0.036
AM37	58	0.883±0.027	0.791±0.043
AS030	60	0.899±0.029	0.818±0.049
AS031	58	0.850±0.029	0.740±0.043
AS043	67	0.905±0.027	0.828±0.043
Mean	62.364	0.874±0.031	0.780±0.048

Table 3.2: Metric results for UNet 2.5D

### 3.2.3. UNet-LSTM

Table 3.3 reports the evaluation metrics of the third model, UNet Lstm, on the validation set. Similarly to the previous models, the performance was evaluated on 11 patients, each consisting of multiple images. The number of images per patient varies between 39 and 90.

The average Dice coefficient over all patients is 0.864, with a standard deviation of 0.040. The IoU metric has a mean value of 0.765, with a standard deviation of 0.060. Looking at individual patients, the model achieved a Dice coefficient ranging from 0.766 to 0.901, and an IoU ranging from 0.624 to 0.820. The highest performance was obtained on patient AS043, with a Dice coefficient of 0.901 and an IoU of 0.820.

Compared to the previous models, UNet Lstm generally performed slightly worse in terms of the mean Dice coefficient, but still achieved satisfactory results. The standard deviation of the Dice coefficient was also slightly higher than in the other models, indicating a larger variability in performance across patients. However, the IoU metric showed a similar level of performance compared to the other models, with a mean value around 0.76-0.82.

Patient	N images	Dice	IoU
HCM29	56	0.894±0.064	0.814±0.094
HCM27	68	0.863±0.037	0.761±0.056
HCM05	73	0.765±0.059	0.624±0.079
HCM10	70	0.876±0.031	0.780±0.048
HCM30	47	0.883±0.032	0.792±0.051
AM08	90	0.864±0.038	0.762±0.059
AM01	39	0.846±0.036	0.734±0.054
AM37	58	0.846±0.066	0.739±0.089
AS030	60	0.899±0.027	0.818±0.045
AS031	58	0.870±0.026	0.771±0.040
AS043	67	0.900±0.025	0.820±0.041
Mean	62.364	0.864±0.040	0.765±0.060

Table 3.3: Metric results for UNet LSTM

### 3.2.4. Model comparison

As seen in Figure 3.2, the global performances of the three models in the validation set can be compared using the Dice coefficient. The UNet 2.5D model achieved the highest average Dice score of 0.874, followed by the UNet LSTM model with an average Dice score of 0.864, and the original UNet model with an average Dice score of 0.864. These results are consistent with the patient-wise performance metrics shown in Tables 3.1, 3.2, and 3.3, where the UNet 2.5D model achieved the highest mean Dice scores for all patients except for AS031

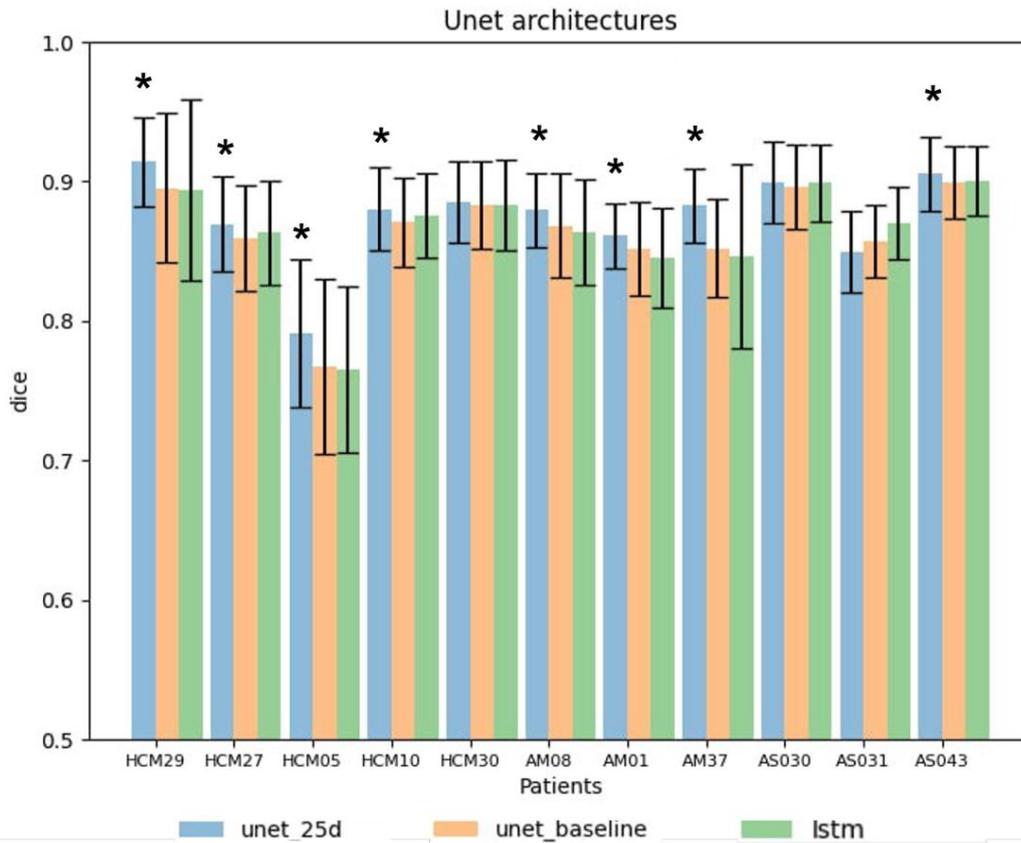


Figure 3.2: Global performances of the 3 models in the validation set. The asterisk (\*) represents the statistical significant model tested by Wilcoxon test ( $p < 0.05$ ). The values for the global flatten models have the following results : U-statistic: 3.0162, p-value: 0.0026

In addition to overall performance metrics, we also evaluated the segmentation performance for each slice of each patient. Figure 3.3 shows a scatter plot of the dice coefficient for the three models across all slices in the validation set. The x-axis represents the number of slices for each patient, while the y-axis represents the dice coefficient. Each point represents the performance of one patient, and the color indicates the model used: blue for UNet, orange for UNet 2.5D, and green for UNet Lstm. We can observe that there is a clear trend at the end of the plot, where the performance of the UNet 2.5D model is consistently better than the other two models. However, it is worth noting that there is considerable variability in the performance across patients.

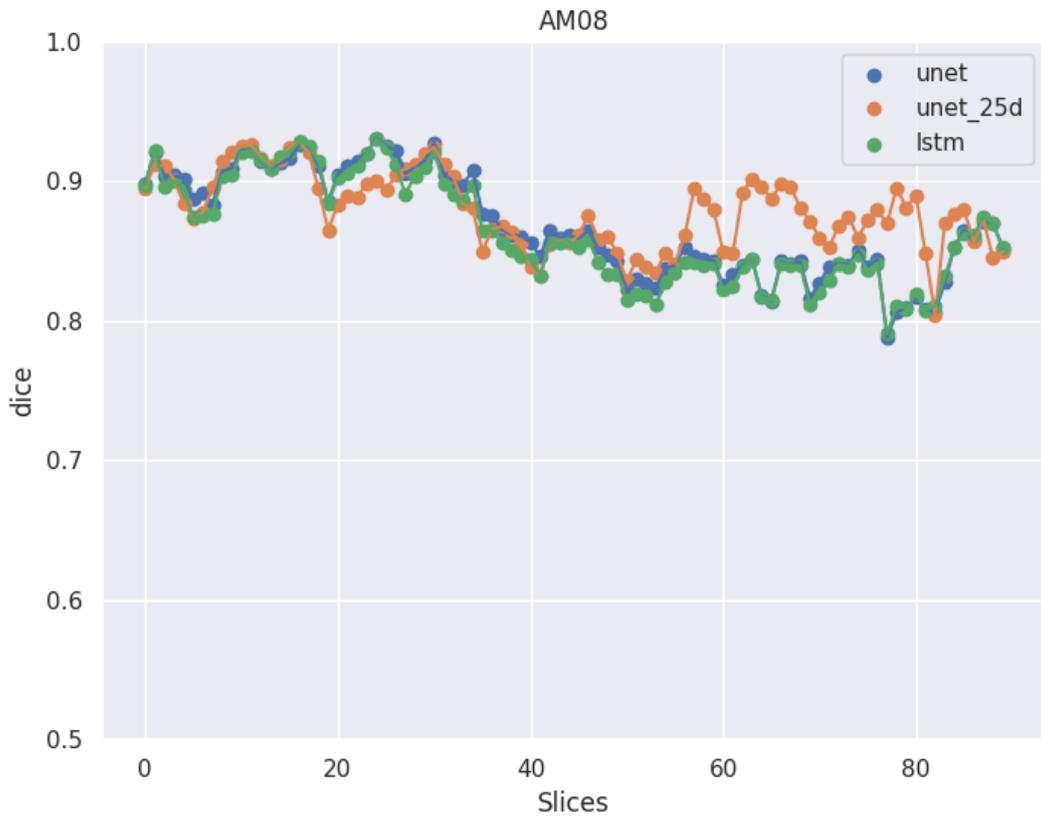


Figure 3.3: Dice coefficient for each slice of the patient AM08. In this patient we can see how the UNet 2.5D performs significantly better in the last slices

### 3.2.5. Visual inspection

In addition to the quantitative metrics obtained from the different models, we also performed a manual inspection of the images and the corresponding masks to have a visual understanding of the performance of each model. This qualitative evaluation provides a complementary insight into the models' performance and can help in identifying areas of improvement. In this section, we present some of the results of the manual inspection, focusing on the differences between the models and highlighting their strengths and weaknesses.

The figure 3.4 shows a comparison between the ground truth masks and the predicted masks for a particular patient (AM08) from the different models. Specifically, the figure displays the masks generated by UNet, UNet 2.5D, and UNet LSTM models. The purpose of this figure is to visually compare the segmentation performance of the three models on this particular patient.

From the figure, we can see that the UNet 2.5D model outperforms the other two models in terms of segmentation accuracy. In particular, we can observe that the masks generated by the UNet 2.5D model are very close to the ground truth masks, while the other two models have some inaccuracies, especially on the top part of the masks. This can be seen as an artifact that is present in the other models but not in the UNet 2.5D model.

Overall, this figure provides evidence that the UNet 2.5D model has better segmentation performance compared to the other two models, at least for this particular patient.

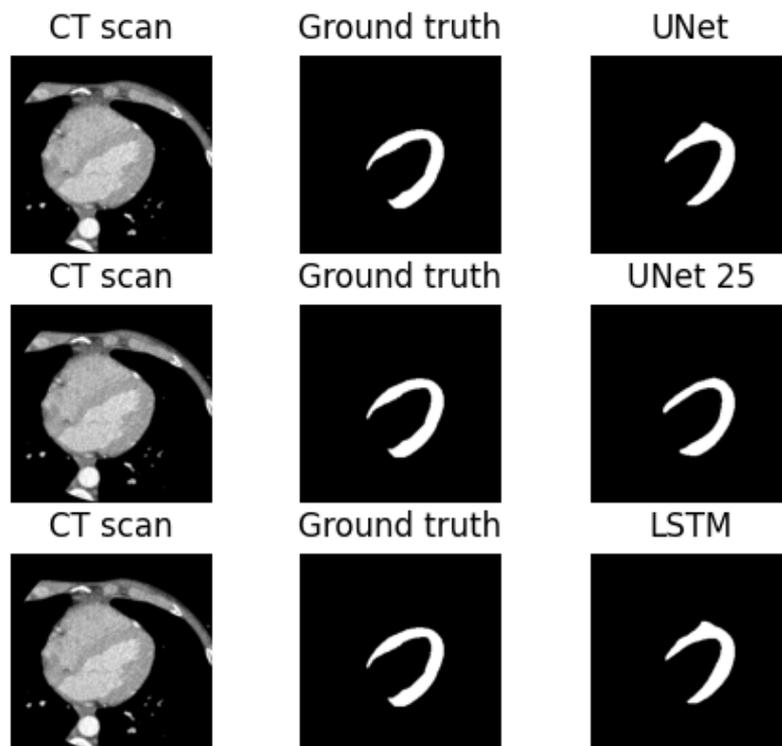


Figure 3.4: In this figure we can see the masks from the different models. We can clearly see that the UNet 2.5D outperforms all the other models

The behaviour is different if we look at another CT scan, in particular for patient AS030. As we can see in Figure 3.5, all three models produce masks that are similar to the ground truth. The performance of each model is comparable and there is no clear winner in this case. We inspected the images manually to have a visual clue of the segmentation output, and it is clear that all models perform well on this scan. This indicates that the performance of the models can vary depending on the patient and the specific characteristics of

the heart images.

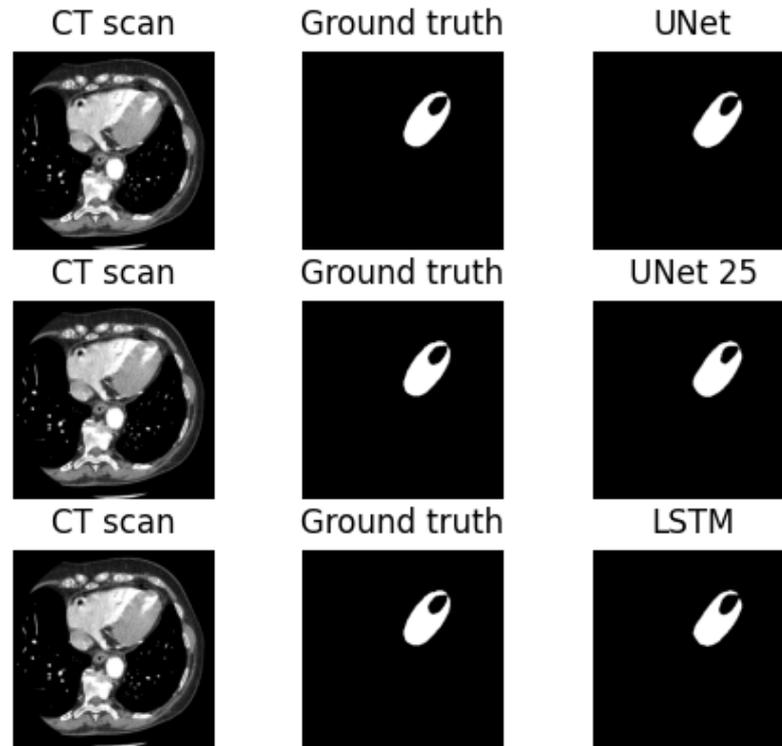


Figure 3.5: In this figure we can see the masks from the different models. We can clearly see that the no model outperforms the others

### 3.3. Test set

In this section, we present the evaluation results of our models on the test set. We use the best performing model from the validation set, which is the UNet 2.5D, to evaluate the segmentation performance on the test set.

The segmentation results for the studied patients are summarized in Table 3.4. The table includes the patient ID, number of images, mean and standard deviation of Dice coefficient, and mean and standard deviation of Intersection over Union (IoU) coefficient. From the table, we can see that the models achieved high performance, with mean Dice coefficients ranging from 0.877 to 0.912 and mean IoU coefficients ranging from 0.782 to 0.839.

Overall, the results demonstrate the effectiveness of the segmentation models in accurately

segmenting the studied patients' images. The high mean Dice and IoU coefficients indicate that the models were able to accurately segment the left ventricle, which is critical for diagnosing and treating heart disease. The additional visualizations of the masks and contour provide a better understanding of the models' performance and can help guide future improvements in segmentation algorithms.

Patient	N images	Dice	IoU
HCM13	75	$0.869 \pm 0.111$	$0.776 \pm 0.149$
HCM01	64	$0.907 \pm 0.021$	$0.831 \pm 0.035$
AM29	115	$0.900 \pm 0.031$	$0.819 \pm 0.050$
AS054	54	$0.890 \pm 0.020$	$0.802 \pm 0.032$
AS002	71	$0.912 \pm 0.012$	$0.839 \pm 0.021$
AS021	40	$0.885 \pm 0.017$	$0.794 \pm 0.027$
AS024	75	$0.877 \pm 0.031$	$0.782 \pm 0.048$
Mean	70.571	$0.891 \pm 0.0345$	$0.806 \pm 0.051$

Table 3.4: Summary of segmentation results for the studied patients.

### 3.3.1. Global metrics

To gain further insight into the segmentation performance, we conducted a more in-depth analysis of the metrics for a single patient, AS054. The results of this analysis are presented in Figure 3.6, which shows the segmentation performance for this patient as a plot of the Dice coefficient versus the number of slices. As can be seen, the Dice coefficient remains consistently high across all the slices, indicating that the segmentation model was able to accurately segment the myocardium throughout the entire volume.

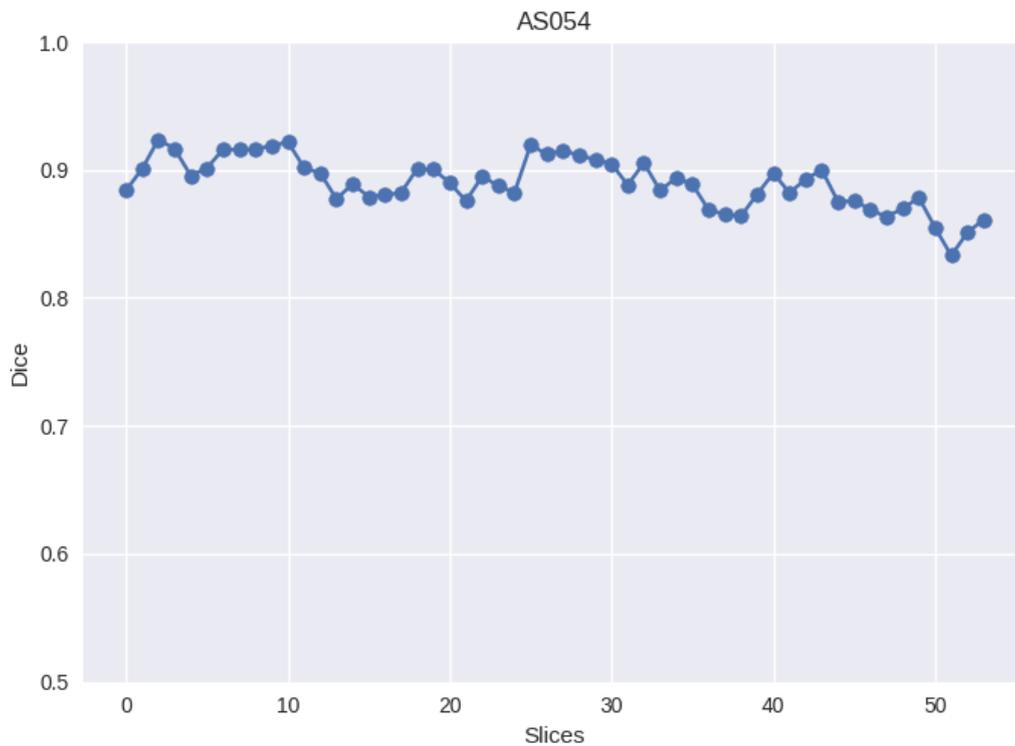


Figure 3.6: The plot shows the segmentation performance for patient AS054, represented as a plot of the Dice coefficient versus the number of slices. The Dice coefficient remains consistently high across all the slices, indicating that the segmentation model was able to accurately segment the LV throughout the entire volume.

In this plot, shown in Figure 3.7, we inspect the segmentation performance for patient AS021. Similarly to the previous plot, we visualize the Dice coefficient across all the slices of the CT volume. The Dice coefficient remains consistently high across the slices, indicating that the segmentation model was able to accurately segment the LV throughout the entire volume. However, we notice a slight trend towards worse performance at the start and at the end of the volume. Nonetheless, the overall performance for this patient is still good and comparable to the other patients in the dataset.

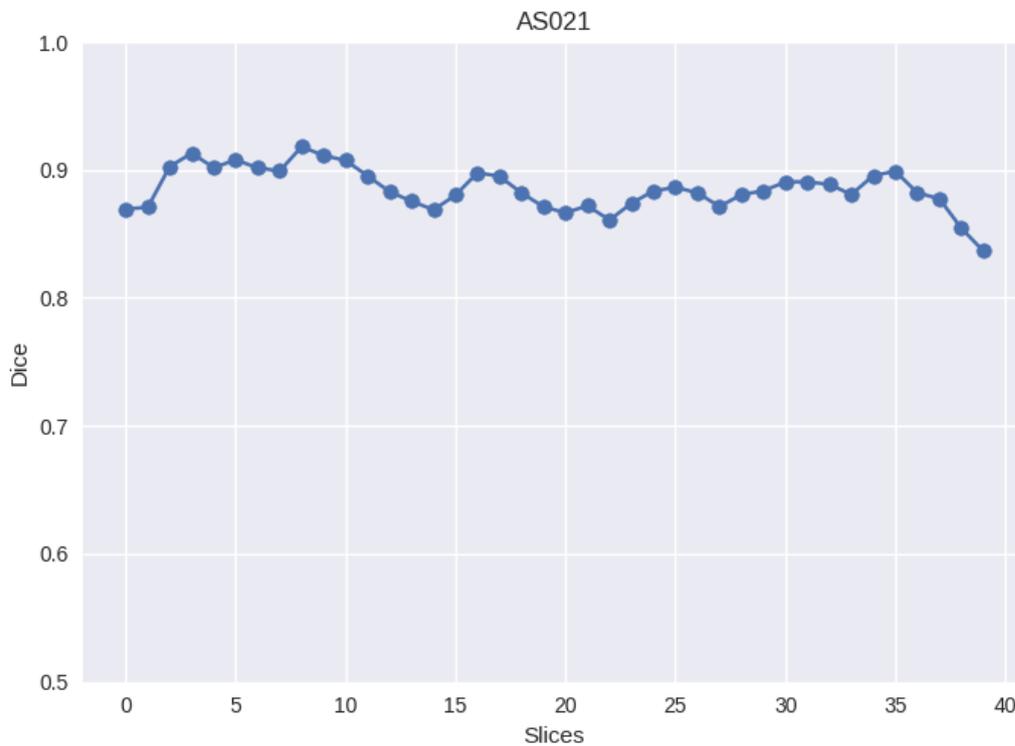


Figure 3.7: The plot shows the segmentation performance for patient AS021, represented as a plot of the Dice coefficient versus the number of slices.

### 3.3.2. Visual inspection

In addition to the quantitative metrics obtained from the model, a manual inspection of the images and the corresponding masks was performed to gain a better understanding of the model's performance on the test set. This qualitative evaluation offers a complementary insight into the models' performance and can identify areas for improvement.

Figure 3.8 shows a comparison between the original CT scan, the ground truth mask, and the predicted mask for patient AS054. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask. The comparison indicates that the predicted mask is very similar to the ground truth mask, indicating that the segmentation model was able to accurately segment the LV for this patient.

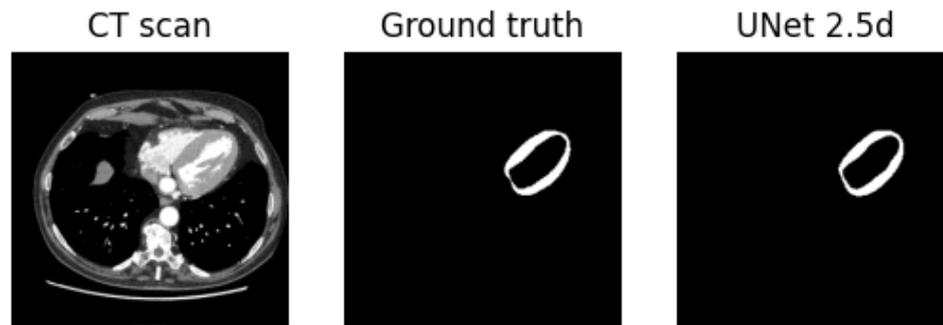


Figure 3.8: Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient AS054. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask.

In addition to the image in Figure 3.8, we also created a comparison between the original CT scan, the ground truth, and the predicted contours overlaid on top for patient AS054, as shown in Figure 3.9. The ground truth contours are displayed in blue, while the predicted contours are in yellow. This image provides an even clearer comparison between the ground truth and predicted masks and highlights the high accuracy achieved by the segmentation model for this patient.

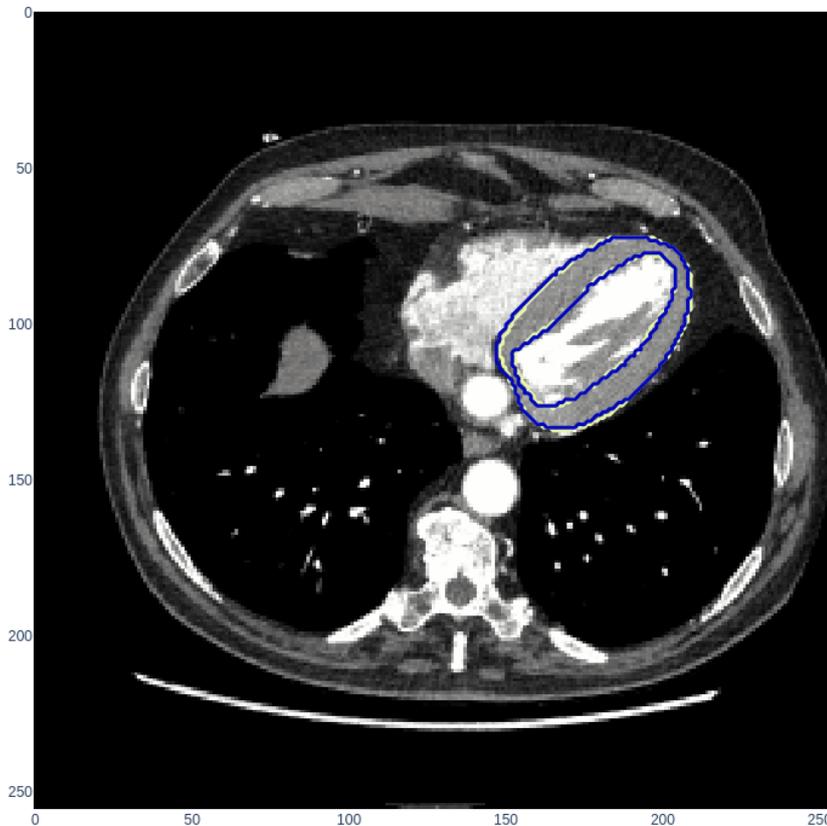


Figure 3.9: Comparison between the original CT scan with the ground truth and predicted contours overlaid on top. The ground truth is blue and the prediction is yellow. The segmentation model was able to accurately segment the LV for this patient

We present the evaluation results for patient another patient HCM13. Figure 3.10 shows a comparison between the original CT scan, the ground truth mask, and the predicted mask. As we can see, there are some differences between the ground truth and the predicted mask, with the latter having a slightly different shape. To further investigate these differences, we also overlaid the ground truth and predicted LV contours on top of the original CT scan, as shown in Figure 3.11. The blue contour represents the ground truth, while the yellow contour represents the predicted segmentation.

Upon closer inspection of the original image, it appears that the left ventricle is better segmented by the model compared to the ground truth. This suggests that the model was able to accurately capture the anatomical features of the patient, and that the differences in contour shape may be due to individual variations in anatomy or imaging artifacts.

Overall, these results demonstrate that the model can achieve accurate segmentation of the LV in challenging cases, even in the presence of anatomical variations or imaging artifacts

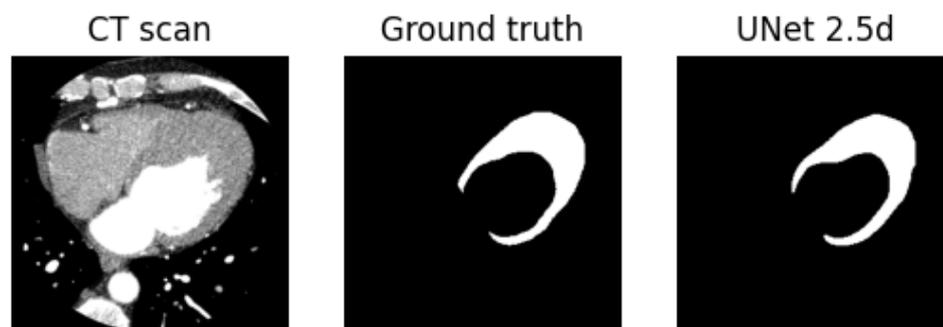


Figure 3.10: Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient HCM13. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask

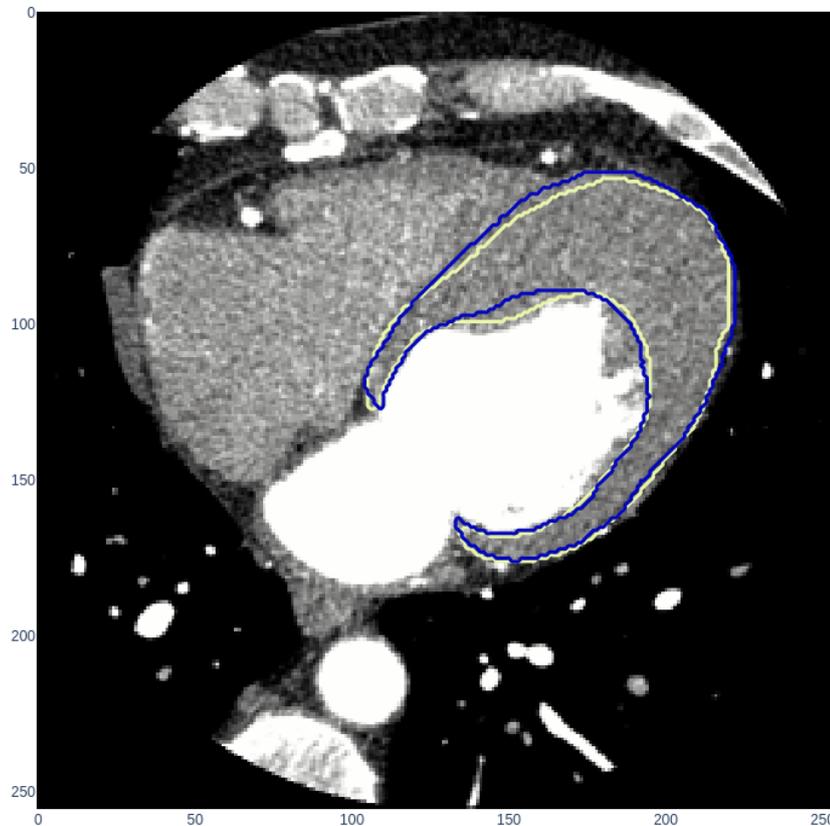


Figure 3.11: Comparison between the original CT scan for patient HCM13, with the ground truth and predicted LV contours overlaid on top. The blue contour represents the ground truth, while the yellow contour represents the predicted segmentation.

The segmentation performance of patient AS021 is shown in Figure 3.12. The comparison between the original CT scan, the ground truth mask, and the predicted mask is shown in the figure. It can be observed that the predicted mask is quite similar to the ground truth mask, indicating that the segmentation model was able to accurately segment the LV for this patient. However, upon closer inspection of the original image, it can be seen that the predicted segmentation has a slightly different shape compared to the ground truth. Despite this, it appears that the left ventricle is better segmented by the model compared to the ground truth, which suggests that the model was able to accurately capture the anatomical features of the patient.

The segmentation performance of patient AS021 is also shown in Figure 3.13. The figure shows the comparison between the original CT scan for patient AS021, with the ground

truth and predicted LV contours overlaid on top.



Figure 3.12: Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient AS021. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask

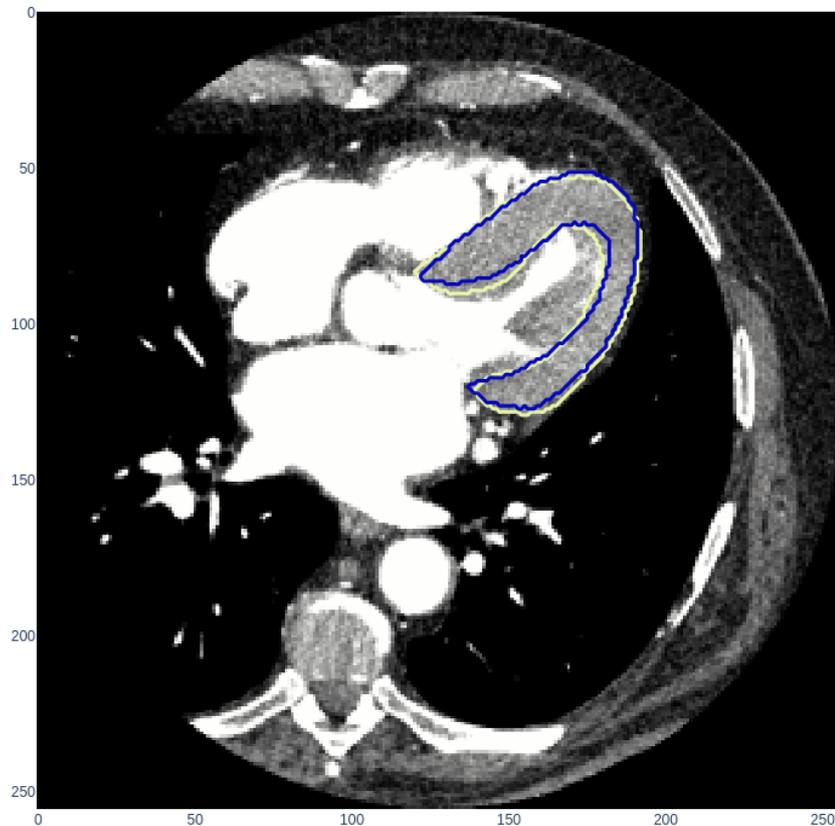


Figure 3.13: Comparison between the original CT scan for patient AS021, with the ground truth and predicted LV contours overlaid on top. The blue contour represents the ground truth, while the yellow contour represents the predicted segmentation. The comparison indicates that there is a difference in the shape of the contours, with the predicted segmentation appearing to have a slightly different shape compared to the ground truth. However, upon closer inspection of the original image, it appears that the left ventricle is better segmented by the model compared to the ground truth. This suggests that the model was able to accurately capture the anatomical features of the patient, and that the differences in contour shape may be due to individual variations in anatomy or imaging artifacts.

# 4 | Discussion

The results obtained in this study demonstrated the potential of deep learning models for the segmentation of the left ventricle in cardiac CT images. The performance of the models were evaluated using quantitative metrics such as the Dice coefficient and Intersection over Union coefficient, as well as visual inspection of the segmentation results.

## 4.1. Literature review

Image segmentation plays a crucial role in the field of medical imaging, particularly in the analysis of cardiac structures like the left ventricle (LV). Accurate segmentation of the LV is essential for diagnosing and monitoring various cardiovascular diseases. Over the years, numerous approaches have been proposed for the segmentation of the LV in computed tomography (CT) scans.

### Threshold-based methods

Threshold-based segmentation methods are the most straightforward approach to segment the LV. These methods rely on the intensity values of voxels in the CT image to separate the LV from other structures.

One of the most popular threshold-based methods is Otsu's method [32], which seeks an optimal threshold value to minimize the intraclass variance. Despite its simplicity, this method often yields unsatisfactory results due to the heterogeneity of CT image intensities.

### Region-based methods

Region-based methods rely on the homogeneity of regions in the image to segment the LV.

Region growing is a popular technique that starts with seed points and iteratively expands the region by including neighboring voxels based on predefined criteria [33]. However, this

method is sensitive to the initial seed points and noise in the image.

## Edge-based methods

Edge-based methods aim to identify the boundaries of the LV by detecting discontinuities in intensity values.

Active contours, also known as snakes, are a widely used edge-based method [34]. These methods iteratively evolve a curve to fit the LV boundary by minimizing an energy function. However, they often struggle with the initialization and convergence to local minima.

## Machine learning-based methods

Machine learning-based methods have gained popularity in recent years due to their ability to learn complex patterns in the data.

DL, particularly convolutional neural networks (CNNs), have achieved state-of-the-art results in LV segmentation [35]. These models learn hierarchical feature representations from large annotated datasets and can generalize well to unseen data. However, they require significant computational resources and large amounts of annotated data for training.

### V-Net

Milletari et al. [36] proposed the V-Net, a 3D CNN architecture designed for volumetric medical image segmentation. The authors reported a Dice coefficient of 0.91 on the Prostate dataset. The V-Net has been adapted for LV segmentation tasks in CT scans, showing promising results.

### DeepLabv3+

Chen et al. [37] introduced DeepLabv3+, an encoder-decoder architecture that combines atrous spatial pyramid pooling (ASPP) and a fully connected conditional random field (CRF). They achieved a mean Dice coefficient of 0.80 on the PASCAL VOC 2012 dataset. Although this study focused on natural images, DeepLabv3+ has been adapted for LV segmentation in CT scans, yielding competitive results.

### U-Net-LSTM

Poudel et al. [38] proposed the Recurrent Fully Convolutional Network (RFCN), which combines U-Net and LSTM for the segmentation of cardiac structures, including the left

ventricle, in cardiac magnetic resonance (MR) images. The authors reported a mean Dice coefficient of 0.89 on the Left Ventricle Segmentation Challenge dataset. Although this study focused on MR images, the U-Net-LSTM architecture can be adapted for LV segmentation in CT scans. We used a variant of the UNet-LSTM achieving comparable results with a dice score of 0.864

## 4.2. UNet 2.5D

The validation results showed that the best-performing model was the 2.5D U-Net model, which achieved a mean Dice coefficient of 0.874 and a mean Intersection over Union coefficient of 0.780. The other two models, U-Net and U-Net with LSTM, also achieved good performance, with mean Dice coefficients ranging from 0.877 to 0.898 and mean Intersection over Union coefficients ranging from 0.782 to 0.820. The final performances of the best model, the UNet 2.5D, on the test set are a mean Dice coefficient of 0.891 and a mean Intersection over union of 0.806.

The qualitative evaluation of the segmentation results using visual inspection of the images and corresponding masks revealed that the 2.5D U-Net model produced the most accurate segmentations overall, with the fewest errors and artifacts. However, the other models also produced good results, with few errors or artifacts, indicating that they could be used for clinical applications too.

In addition, the manual inspection of the images and masks provided insights into the strengths and weaknesses of each model. For instance, the U-Net with LSTM model tended to produce masks with thicker boundaries, which may lead to over-segmentation. The 2.5D U-Net model, on the other hand, produced masks with accurate boundaries and was able to capture the anatomical features of the left ventricle with high accuracy in most of the cases.

It is worth noting that the test results confirmed the findings of the validation results, with the 2.5D U-Net model achieving the best performance. The qualitative evaluation of the test results using visual inspection of the images and corresponding masks also confirmed the high accuracy of the 2.5D U-Net model.

Apart from the performance results of the models, there are other factors to consider when choosing the most suitable model for medical image segmentation tasks. One such factor is the complexity and training time of the model.

Indeed one of the main limitations of LSTMs is that they take longer to train compared to other neural network architectures, such as feedforward neural networks. This is because

the recurrent nature of LSTMs requires passing information across time steps, which leads to longer training times. Moreover, LSTMs require more memory to train due to the larger number of parameters.

According to a study by Greff et al. [39], training LSTMs can take up to five times longer than training a feedforward neural network. This time requirement can be particularly problematic when training on large datasets, such as medical images. Another limitation of LSTMs is that they are easy to overfit, especially when the dataset is small. Overfitting occurs when the model learns the noise in the training data instead of the underlying patterns. This can lead to poor performance on new, unseen data.

Furthermore, implementing regularization techniques like dropout in LSTMs is more challenging compared to other neural network architectures. Dropout is a popular regularization technique used to prevent overfitting in neural networks. The dropout technique involves randomly dropping out some of the neurons during training. This forces the network to learn redundant representations, which helps prevent overfitting. However, applying dropout to the recurrent connections in LSTMs requires careful implementation, as noted in a study by Gal and Ghahramani [40].

Therefore, while LSTMs may be suitable for certain medical image segmentation tasks, the UNet 2.5D model may be a better choice for medical image segmentation tasks with large datasets and limited computational resources. In our study, we chose the UNet 2.5D model as it achieved higher accuracy while requiring less training time and fewer computational resources compared to the LSTM-based UNet model.

Overall, the results of this study demonstrate the potential of deep learning models for the segmentation of the left ventricle in cardiac CT images. The findings of this study demonstrate that the proposed segmentation model utilizing the U-Net 2.5D architecture yields comparable Dice scores in left ventricle segmentation from CT scans, when compared with existing models from the literature. To the best of our knowledge, this is the first time the U-Net 2.5D architecture has been applied in the segmentation of the left ventricle using CT scans.

# 5 | Conclusions and future developments

This study has demonstrated the potential of deep learning models for accurate segmentation of the LV in cardiac CT images. The results provide important insights into the strengths and weaknesses of different models for medical image segmentation tasks, which can help researchers and practitioners choose the most suitable model for their specific application. Furthermore, accurate segmentation of the left ventricle is essential for diagnosing and monitoring cardiovascular diseases, making the potential of these models particularly promising for clinical applications.

## 5.1. Strengths

UNet 2.5D offers a superior understanding of spatial relationships between slices, resulting in more accurate segmentation. This method is particularly advantageous for left ventricle segmentation, where images can be acquired from various medical imaging modalities and conditions.

In comparison to 3D approaches and LSTM, UNet 2.5D has lower computational requirements. The employment of simple convolutions facilitates faster processing and reduced memory consumption, rendering it more efficient and appropriate for real-time applications.[41]

Additionally, the UNet 2.5D architecture supports easy implementation of transfer learning, allowing the network to capitalize on pre-trained models from related tasks to enhance performance [42]. This feature contributes to decreased training time and better generalization across diverse datasets.

## 5.2. Limitations

While the model have demonstrated promising results, it is essential to consider its limitations when interpreting its performance and generalizability. We discuss some limitations of our study.

We have not tested our model on publicly available datasets, which makes it difficult to directly compare its performance with other models in the literature that use the same data.

Also as with most studies on CT scan segmentation, the images typically come from the same scanners. In real-world situations, we may encounter cases where we need to test the model with new scanners, which could potentially decrease its performance. This indicates that our model may not be as robust to generalization as desired.

Finally the manual annotations on the datasets can vary in terms of the criteria used for segmenting the LV, as different expert radiologists were involved in the annotation process. This inconsistency may affect the quality of the ground truth labels and, in turn, the performance of the segmentation models.

## 5.3. Future works

Looking ahead, there are several areas for future research in medical image segmentation.

Investigation of ensembles of different models to further improve segmentation accuracy could be fundamental. This approach has shown promise in other domains.

Additionally, it will be important to explore the transferability of the models to different imaging modalities and patient populations. This could expand the applicability of these models to a wider range of clinical settings.

Another way to go are Post-processing techniques such as Markov random fields and active contours snake that could be explored to refine segmentation results. MRF can be used to model the spatial dependencies between neighboring pixels, while active contours snake can be used to iteratively adjust the contour shape to minimize an energy function. These techniques have shown promise in improving segmentation results in other applications and could potentially be applied to medical image segmentation.

Finally, it will be crucial to assess the clinical feasibility and impact of using DL models for medical image segmentation tasks. This could involve conducting clinical trials to evaluate the impact of using these models on patient outcomes such as diagnosis, treat-

ment planning, and prognosis. Ensuring the safety and efficacy of these models in clinical settings will be essential to their successful implementation.

### Conclusion

In summary, our study demonstrated the potential of deep learning models for accurate segmentation of the left ventricle in cardiac CT images. Our findings provide important insights into the strengths and weaknesses of different models for medical image segmentation tasks, and open up several avenues for future research in this area.



## Bibliography

- [1] X. Zhu, “Semi-supervised learning literature survey,” *Technical Report 1530, Department of Computer Sciences, University of Wisconsin-Madison*, 2005.
- [2] X. Liu, L. Song, S. Liu, and Y. Zhang, “A review of deep-learning-based medical image segmentation methods,” *Sustainability*, vol. 13, no. 3, 2021. [Online]. Available: <https://www.mdpi.com/2071-1050/13/3/1224>
- [3] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.
- [4] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [5] F. Chollet, *Deep Learning with Python*. Simon and Schuster, 2021.
- [6] B. Ait Skourt, A. El Hassani, and A. Majda, “Lung ct image segmentation using deep neural networks,” *Procedia Computer Science*, vol. 127, pp. 109–113, 2018.
- [7] A. Paszke, A. Chaurasia, S. Kim, and E. Cukurciello, “Enet: A deep neural network architecture for real-time semantic segmentation,” *Arxiv*, June 2016.
- [8] A. Klein, M. Willner, L. Schröder, A. Graser, A. Cavallaro, K. Maier-Hein, M. Kirschner, M. Lell, H.-U. Kauczor, and M. Härtling, “Automatic bone segmentation in whole-body ct images,” *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, no. 1, pp. 21–29, 2018.
- [9] S. Sun, Z. Cao, H. Zhu, and J. Zhao, “A survey of optimization methods from a machine learning perspective,” *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3668–3681, 2020.
- [10] Z. T. Al-Sharify, T. A. Al-Sharify, N. T. Al-Sharify, and H. Y. Naser, “A critical review on medical imaging techniques (ct and pet scans) in the medical field,” *IOP Conference Series: Materials Science and Engineering*, vol. 870, no. 1, p. 012043, June 2020. [Online]. Available: <https://doi.org/10.1088/1757-899x/870/1/012043>

- [11] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer, 2015, pp. 234–241, arXiv preprint arXiv:1505.04597. [Online]. Available: <https://app.dimensions.ai/details/publication/pub.1017774818>
- [12] A. Vidhya, “All you need to know about skip connections,” 2021. [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/08/all-you-need-to-know-about-skip-connections/>
- [13] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, “Review of deep learning: concepts, cnn architectures, challenges, applications, future directions,” *Journal of Big Data*, vol. 8, 2021.
- [14] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, “U-net and its variants for medical image segmentation: A review of theory and applications,” *IEEE Access*, vol. 9, pp. 82 031–82 057, 2021.
- [15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [16] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.
- [17] C. Olah, “Understanding lstm networks,” <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>, 2015.
- [18] P. T. Inc., “Dash documentation,” <https://dash.plotly.com/>, accessed 30 Mar. 2023.
- [19] G. Dclunie, “Dicom—a standard for digital imaging,” *Journal of digital imaging*, vol. 12, no. 2 Suppl, pp. 3–10, 1999.
- [20] T. P. Developers, “pydicom: Pydicom,” 2021.
- [21] R. C. Gonzalez and R. E. Woods, *Digital image processing*. Prentice Hall, 2008.
- [22] E. K. Fishman, D. R. Ney, D. G. Heath, F. M. Corl, K. M. Horton, P. T. Johnson, and J. S. Babb, “Ct of the heart and great vessels,” *Journal of thoracic imaging*, vol. 20, no. 1, pp. 33–50, 2005.
- [23] S. D. Goldstein, S. E. Seltzer, K. E. Muller, H. K. Muller, Y. Z. Lee, H. Chen, V. R. Doppalapudi, K. Nguyen, and A. Khademhosseini, “Optimizing ct display parameters

- for enhancement of the pulmonary artery: a comparison of three contrast injection rates,” *Academic radiology*, vol. 14, no. 4, pp. 455–463, 2007.
- [24] A. Buslaev, V. Iglovikov, E. Khvedchenya, and A. Parinov, “Albumentations: fast and flexible image augmentations,” <https://github.com/albumentations-team/albumentations>, 2020.
- [25] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.
- [26] F. Milletari, N. Navab, and S.-A. 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.
- [27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [28] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.
- [29] D. Miceli Pranio, “Mri and ct segmentation with deep learning using multi-slice inputs,” Master’s Thesis, Politecnico di Milano, Milan, Italy, July 2021, student ID: 928262. [Online]. Available: <https://www.politesi.polimi.it/handle/10589/152722>
- [30] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, vol. 9351, pp. 234–241, 2015.
- [31] C. Chen, C. Qin, H. Qiu, G. Tarroni, J. Duan, W. Bai, and D. Rueckert, “Deep learning for cardiac image segmentation: A review,” *Frontiers in Cardiovascular Medicine*, vol. 7, p. 25, 2020.
- [32] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [33] R. Adams and L. Bischof, “Seeded region growing,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641–647, 1994.
- [34] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.
- [35] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian *et al.*,

- “A survey on deep learning in medical image analysis,” *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.
- [36] F. Milletari, N. Navab, and S.-A. 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.
- [37] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 801–818.
- [38] R. P. K. Poudel, P. Lamata, and G. Montana, “Recurrent fully convolutional neural networks for multi-slice mri cardiac segmentation,” in *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2016, pp. 740–744.
- [39] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, pp. 2222–2232, October 2017.
- [40] Y. Gal and Z. Ghahramani, “A theoretically grounded application of dropout in recurrent neural networks,” in *Advances in Neural Information Processing Systems*, 2016, pp. 1019–1027.
- [41] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:2001.06268*, 2020.
- [42] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, “Convolutional neural networks for medical image analysis: Full training or fine tuning?” *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

# A | Appendix A



## List of Figures

1.1	Visualization of the features from different layers of a CNN. In CNNs, the lower layers typically detect simpler features such as edges and corners, while the higher layers learn more complex features such as shapes and textures. Different filters or kernels in a layer may detect edges of different shapes, orientations, and sizes, depending on their weights and parameters. For example, a filter with a vertical orientation may detect vertical edges, while a filter with a diagonal orientation may detect diagonal edges . . . .	5
1.2	Different pooling layers. Max and average pooling. . . . .	6
1.3	ReLU function plot in 2d axis. . . . .	7
1.4	Example of a Ct scan slice from a patient. . . . .	10
1.5	UNet original architecture. . . . .	11
1.6	LSTM architecture . . . . .	13
2.1	Barplot that represents the number of patients for each pathology. . . . .	15
2.2	Pie chart that represents the percentage of images per each pathology . . .	16
2.3	Number of slices having the manual target segmentation per patient . . . .	17
2.4	Comparison of the preprocessing. On the left we can see the original image in which is hard to distinguish the ventricle. On the right instead is much easier to distinguish it . . . . .	20
2.5	Visual representation of the IOU. Overlap over union. . . . .	24
2.6	Visual representation of the Dice coefficient. . . . .	24
2.7	Basic UNet architecture. . . . .	27
2.8	Different input/output for the UNet, UNet2.5D . . . . .	28
2.9	Example of UNet-lstm architecture from [31] . . . . .	30
2.10	The GUI allows selecting a patient and the segmentation models' masks to display. A slider enables browsing through the slices of the selected patient.	31
3.1	This picture represents the modification applied by the post processing. We can see that a second component is correctly deleted by the post processing.	34

3.2	Global performances of the 3 models in the validation set. The asterisk (*) represents the statistical significant model tested by Wilcoxon test ( $p < 0.05$ ). The values for the global flatten models have the following results : U-statistic: 3.0162, p-value: 0.0026 . . . . .	38
3.3	Dice coefficient for each slice of the patient AM08. In this patient we can see how the UNet 2.5D performs significantly better in the last slices . . .	39
3.4	In this figure we can see the masks from the different models. We can clearly see that the UNet 2.5D outperforms all the other models . . . . .	40
3.5	In this figure we can see the masks from the different models. We can clearly see that the no model outperforms the others . . . . .	41
3.6	The plot shows the segmentation performance for patient AS054, represented as a plot of the Dice coefficient versus the number of slices. The Dice coefficient remains consistently high across all the slices, indicating that the segmentation model was able to accurately segment the LV throughout the entire volume. . . . .	43
3.7	The plot shows the segmentation performance for patient AS021, represented as a plot of the Dice coefficient versus the number of slices. . . . .	44
3.8	Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient AS054. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask. . . . .	45
3.9	Comparison between the original CT scan with the ground truth and predicted contours overlaid on top. The ground truth is blue and the prediction is yellow. The segmentation model was able to accurately segment the LV for this patient . . . . .	46
3.10	Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient HCM13. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask . . . . .	47
3.11	Comparison between the original CT scan for patient HCM13, with the ground truth and predicted LV contours overlaid on top. The blue contour represents the ground truth, while the yellow contour represents the predicted segmentation. . . . .	48
3.12	Comparison between the original CT scan, the ground truth mask, and the predicted mask for patient AS021. The leftmost image shows the original CT scan, the middle image shows the ground truth mask, and the rightmost image shows the predicted mask . . . . .	49

3.13 Comparison between the original CT scan for patient AS021, with the ground truth and predicted LV contours overlaid on top. The blue contour represents the ground truth, while the yellow contour represents the predicted segmentation. The comparison indicates that there is a difference in the shape of the contours, with the predicted segmentation appearing to have a slightly different shape compared to the ground truth. However, upon closer inspection of the original image, it appears that the left ventricle is better segmented by the model compared to the ground truth. This suggests that the model was able to accurately capture the anatomical features of the patient, and that the differences in contour shape may be due to individual variations in anatomy or imaging artifacts. . . . . 50



## List of Tables

3.1	Metric results for UNet . . . . .	35
3.2	Metric results for UNet 2.5D . . . . .	36
3.3	Metric results for UNet LSTM . . . . .	37
3.4	Summary of segmentation results for the studied patients. . . . .	42



## List of Symbols

<b>Variable</b>	<b>Description</b>
ReLU	rectified linear unit
softmax	softmax function
CE	cross-entropy loss
SGD	stochastic gradient descent
Adam	Adam optimizer
CNN	convolutional neural network
RNN	recurrent neural network
LSTM	long short-term memory
UNet	UNet architecture
FCN	fully convolutional network
LCEA	Largest Component Extraction Algorithm
CT	computed tomography
DL	Deep learning

