Automatic 3D Segmentation in CT images of Congenital Heart Defects using Deep Learning

Matilda Dahlström

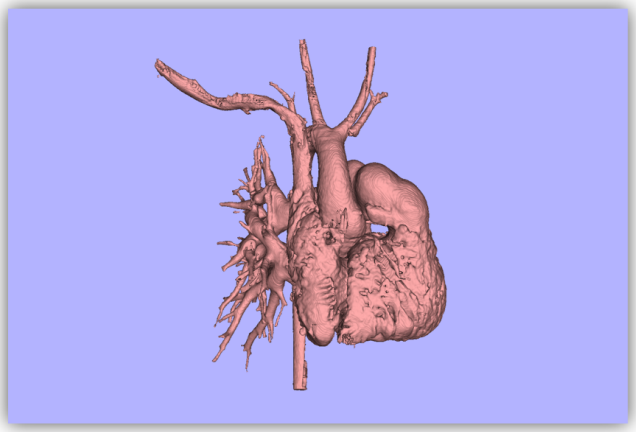
Master's thesis
2022:E25

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

Automatic 3D Segmentation in CT images of Congenital Heart Defects using Deep Learning



Matilda Dahlström

20th May 2022

Abstract

3D segmentations of hearts with congenital heart defects are routinely used today. They are used to study hearts and to prepare before surgery which makes them an important part of patient care. The 3D segmentations are usually created manually, which is a time-consuming process. By automating this process, less time can be spent creating these models, and more time can be spend on patient care.

The purpose of this project was to create a fully automatic method that created 3D segmentation of CT images of hearts with congenital heart defect. The aim was to create this method using deep learning, more specifically the U-net structure.

This was done by manually segment images, which were used as ground truth segmentations during network training. The classes used for training were the Blood Pool, Bone and Pericardium, which were segmented separately. The ground truth segmentations were done by the author of this thesis, and evaluated by others. The hyper parameters of the network were adjusted to fit this project and data augmentations was applied to the training data to extend the training data set. Training was done on 21 patients, and testing was done on seven. The class used for performance evaluation was the Blood Pool and the final output from the network was the Cardiac Model, which included segmentation of the Blood Pool and Pericardium.

The final network in this project segmented the Blood Pool, that scored an average Dice score of 93,5% (±2,9) and a Jaccard Index of 88,0% (±5,0), when evaluated on seven test images.

The results of this thesis shows that creating automatic 3D segmentations of congenital heart defects is possible with the use of deep learning. It also shows that even if the automatically created segmentations are imperfect, they can be used clinically to improve the patient care.

Acknowledgements

I would like to show my gratitude to everyone at the 3D center at Lund University Hospital. It has been rewarding and fun to do my master thesis with you and I highly appreciate you sharing your knowledge with me. I would especially like to thank my supervisor Einar Heidberg for the opportunity to work with this project and for his guidance and engagement during the process. I would also like to thank Pia Sjöberg for providing me with the data needed for this project, your contribution is very appreciated. Finally, I would like to thank my family and friends for their support.

Abbrevations

Machine Learning

ML - Machine Learning

DL - Deep Learning

ANN - Artificial Neural Network

CNN - Convolutional Neural Network

ReLU - Rectified Linear Unit

Medical Image Processing

CT - Computed Tomography

Anatomy

CHD - Congenital Heart Defect

VSD - Ventricular Septal Defect

PDA - Patent Ductus Arteriosus

DORV - Double Outlet Right Ventricle

TGA - Transposition of the Great Arteries

PS - Pulmonary Stenosis

PA - Pulmonary Atresia

TCPC - Total Cavopulmonary Connection

PAPVD - Partial Anomalous Pulmonary Venous Drainage

HLHS - Hypoplastic Left Heart Syndrome

ASD - Atrial Septal Defect

MAPCA - Major Aortopulmonary Collateral Arteries

TAPVR - Total anomalous pulmonary venous return

Contents

1 Introduction

Congenital heart defects (CHDs) are heart defects present at birth. Common CHDs are narrow valves, leaky valves and holes in the inside walls of the heart. More severe forms of CHDs are when the vessels or chambers are completely missing, poorly formed or in the wrong place [1]. These defects are usually discovered at the routine ultrasound in pregnancy week 18-20[2]. Discovering CHDs before birth is important so that the best possible conditions can be created for the baby when it is delivered. For some complicated CHDs, where surgery is needed, this means planning the delivery so that it is performed at one of the two pediatric heart surgery centers in Sweden, which are located in Gothenburg and Lund, [2].

Not all heart defects are discovered before birth. Some children only present symptoms later in life. It is therefore important that the healthcare system is aware of possible symptoms of heart defects and have the best tools at hand to study and treat these defects. The pediatric heart surgery center in Lund performs 350 surgeries yearly. Children with CHDs are offered many forms of surgical treatments here, including heart transplantation [3]. At the hospital, multidisciplinary teams work together so that the best possible care can be offered to these children. One important collaboration is the one between the pediatric center and the 3D center [4].

At the 3D center, engineers work with 3D printing and 3D visualisation to improve the healthcare in Skåne. They help surgeons by providing patient-specific 3D models that can be studied before surgery [5], and design and print implants i bio-compatible materials that are implanted in patients permanently, [6].

The incorporation of 3D models in pediatric heart surgery improve patient safety, since surgeons can be better prepared. It also increase patient satisfactory as the models can be used to explain the defects to the affected families, thereby including them more in the care.

At the 3D center, the 3D models are created using the software Segment 3DPrint. This software is provided by the medical imagining company Medviso AB [7]. The software contains all necessary tools for classical image processing in 2D and 3D. It also contains some automatic segmentation methods that have been developed by other students in previous years, such as automatic bone segmentation, [8]. The process of creating 3D models of hearts is however not automated and requires a lot of expertise and time, as they need to be completed manually.

Today, the physicians send medical images of the patient to the 3D center, together with a referral for the desired 3D model. The engineers at the 3D center then manually create the 3D model of the heart and either print the model or create a 3D visualisation for the physicians to study. The aim of this project is to automate the creation of 3D models of hearts, so that this process can be more time-efficient. By making this process more efficient, the collaboration between the departments can be more

efficient, thus making it possible to incorporate the use of 3D models in more patients.

Based on previous work, it was hypothesized that deep learning would be a great method to automatically create 3D models from given medical images. From previous work it was discovered that the U-net is a great neural network structure for these type of tasks and this structure will therefor be used in this project.

In this report, background to machine learning and medical imaging will be presented, followed by the work that has been done. This includes manually delineating images to be used as ground truth when training the neural network and altering the network structure to fit this particular problem. It also includes the process of designing an evaluation method so that the performance of the networks can be evaluated accurately. The resulting network architecture will be presented, together with some automatic segmentations done on real patients. The implementation of this automatic method in the existing software Segment 3DPrint will also be presented to the reader.

2 Purpose

The aim of this project is to create a fully automatic method that takes CT images of children's hearts and creates a 3D segmentation of the heart. The purpose is to create a method that can segment hearts with different types of congenital heart defects.

The automatic segmentation will be created by a convolutional neural network with a specific architecture, called the U-net. The network will be trained to fit this project by providing the network with training data that is manually delineated ground truth segmentations, and by altering hyper parameters of the network.

The purpose of creating automatic 3D segmentations is to improve patient safety at the pediatric heart surgery center at Lund University Hospital, and in the long run, other institutions as well.

3 Theory

3.1 Machine Learning

Machine learning (ML) is the method of letting computer algorithms improve automatically through experience and the use of data, without being explicitly programmed. The human brain gains knowledge and understanding by analyzing input data from the world and making connections. In the same manner, machine learning relies on input data to understand connections. A machine learning algorithm observes data and identifies features so that it later can use those features to identify structures in previously unseen data, and classify the new data.

Computer Vision is a field of machine learning that includes how computers can gain understanding from digital images or videos. The application of Deep Learning (DL) in the field of computer vision has led to great advancements and made it possible to apply computer vision to more scientific areas. In the following section, some introduction to machine learning and deep learning will be given, together with its application in the field of computer vision and medical imaging.

3.1.1 Artificial Neural Networks

An artificial neural network (ANN) is a machine learning model inspired by the biological nervous system. Like the biological neural network, the ANN is an interconnection of nodes, analogous to neuron. The typical structure of an ANN is presented in 3.1. The properties of the nodes, such as the number of inputs and outputs associated with the node and the weight of each input and output, determines how signals are processed throughout the network. Thus determining the properties of the ANN. [9]

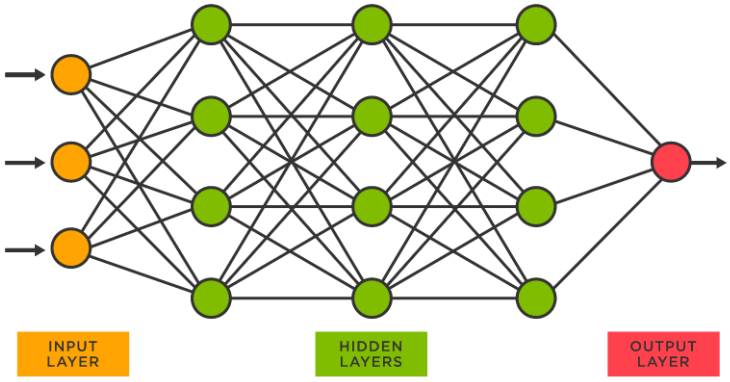


Figure 3.1: Typical structure of an Artificial Neural Network. Each circle is a node and the lines between them are connections with associated weights [10].

The nodes are organized into linear arrays called layers. In the network, there are input layers, hidden layers and output layers. The input to the network can be many

types of data, like images and signals. The number of layers, and nodes in each layer, is determined beforehand based on previous work, and then optimized to fit the project in question through multiple rounds of experiments. If a network is designed with many hidden layers, it is called a deep network, hence the term deep learning. The weights of the ANN are set to some initial values and then updated through a learning process, see more under the chapter 3.1.4, [9].

3.1.2 Convolutional Neural Networks

A convolutional neural network (CNN) is a specialized ANN with similar structure as the traditional ANN. The only difference being the hidden layer, where the CNN has three specialized layers, see Figure 3.2. These are convolutional layers, pooling layers and a fully-connected layer, [11]. A more in-depth explanation of these layers are given in the following sections.

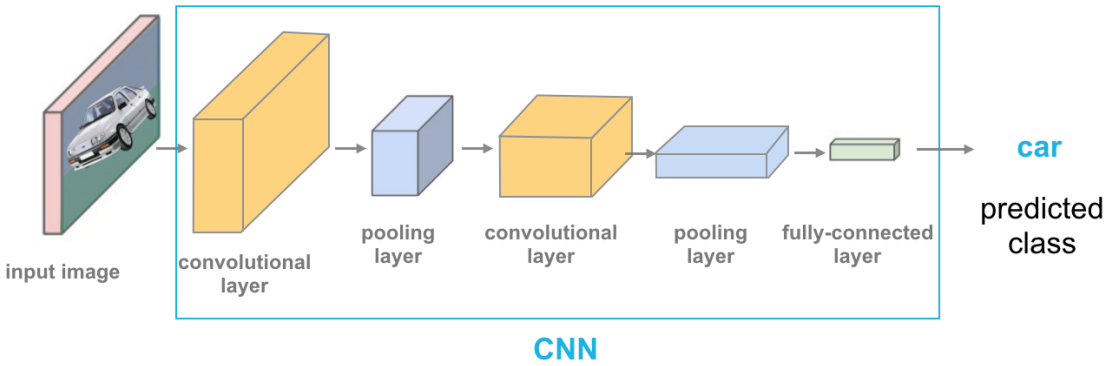


Figure 3.2: Typical structure of a Convolutional Neural Network. The structure of the hidden layer is a combination of convolutional layers, pooling layers and a fully-connected layer [12].

Convolutional layer

The convolutional layers are the core of the CNNs. These layers put the images through a set of convolutional filters, which each activates certain features of the image, [13]. The convolutional filters are 2D arrays of weights, which are updated during the learning process. These filters can vary in size but are typically 3x3 matrices. In each convolutional layer, a convolutional filter is applied to an area of the image and the scalar product is calculated between the input values and the filter values. The product is then fed into an output array, see Figure 3.3. The filter then shifts so that it covers a new area of the image, and computes the scalar product of this area. After the filter has swept across the entire image, a resulting output array has been computed, called the feature map. The feature maps capture interesting features of each image. [14].

After each convolutional layer, an activation function is applied to the feature map. One popular activation function is the Rectified Linear Unit (ReLU) transformation [14].

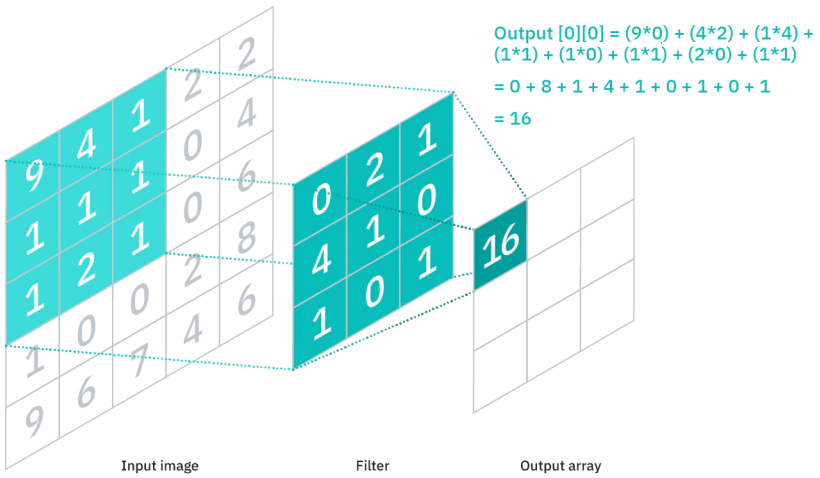


Figure 3.3: Convolution between the input image and the convolutional filter [14].

Pooling layer

The pooling layers exists to reduce the size of the network. Just like the convolutional layer, the pooling layer uses a filter that shifts over the input and computes an output. In the pooling layers a filter, usually of size 2x2, moves across the feature map and selects the pixel with the maximum value of each area as the output. This produces a matrix of smaller size than the original, see Figure 3.4. When a pooling layer is used in a CNN, the input to the pooling layer is the feature map produced by the convolutional layer.[15].

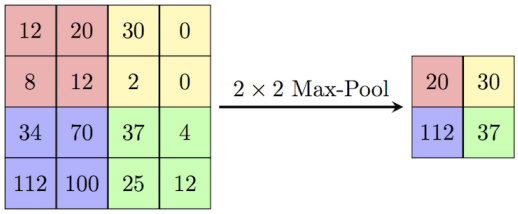


Figure 3.4: Max pooling operation with a filter of size 2x2 [16].

Even though a lot of information is lost in the pooling layers, they are important since they help reduce complexity, improve efficiency and limit the risk of overfitting, [14].

Fully connected layer

The fully connected layer connects each node in the second to last layer directly to nodes in the output layer. In this layer, the final classification takes place. This layer is common in image classification tasks but not in semantic segmentation tasks as the outputs are different, see more under 3.2.1.

3.1.3 U-net

The U-net is a CNN with a structure first developed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox [17]. Because of its robust performance and easy implementation, this network structure has become widely popular, especially for computer vision tasks. The network architecture of the U-net consists of a contracting path where the image is encoded to capture the context, and an expansive path where the image is decoded and precise localisation is made possible. The contextual information and the localisation is then combined to create the final network output. Figure 3.5 shows the U-net architecture. In this specific example, the input is an image and the output is a segmentation.

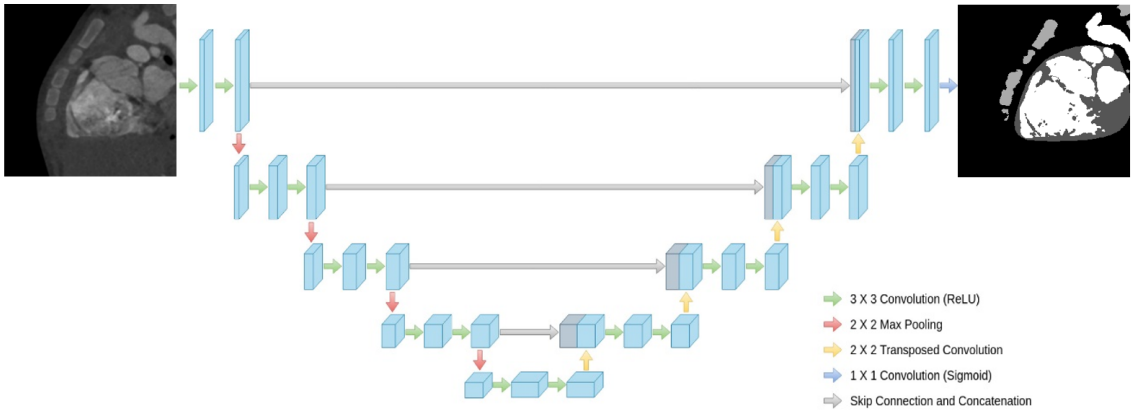


Figure 3.5: U-net structure. The arrows denote the different operations. Image from [18], and adapted by the author.

The contracting path follows the typical architecture of CNNs, with some special adaptions. A common layer combination is the one where two 3x3 convolutional layers are stacked together, which allows for more complex features to be detected, [15]. These convolutional layers are followed by ReLU and a 2x2 max pooling operation. This combination is repeated multiple times in the network,[17]. For each step in the contracting path, the number of feature channels in the network is doubled. These feature channels are used to propagate context information to higher resolution layers, [17].

In the extracting path, the feature map is upsampled to increase its size. This upsampling is followed by an "up-convolution", also called a transposed convolution, with a 2x2 filter to reduce the number of feature channels, and a combination of convolutional layers and ReLU.

At the final layer of the network, where most CNNs have a fully connected layer, the U-net has a specialized layer that uses convolution to map the features detected in the image to a class, thus creating the resulting network output, [17].

When the input to the U-net is an image, and the purpose of the network is semantic image segmentation, the output from the network is also an image. In this image, each pixel has been assigned a label, corresponding to a class, see more under 3.2.1.

The U-net can be used on 2D images as well as 3D images. For 3D images, the image is divided into three orthogonal stacks of 2D images. For each of these 2D stacks, a U-net is applied. The output from each of the three networks, is a segmented 3D volume, but with image stacks in different directions. A pixel-wise voting between the segmented 3D volumes is done, and the final 3D segmentation is created.

3.1.4 Training

When training a neural network, the weights between the nodes are updated to optimize the performance of the network. This learning process can be divided into two main categories; supervised learning and unsupervised learning. In supervised learning, a ground truth is provided. This could be a data set containing training data with inputs and corresponding target outputs. This data is given to the network so that the weights can be adjusted to minimize the error between network output and the ground truth. In unsupervised learning, no ground truth is provided, and the learning process is based on finding underlying patterns from the input data alone. The appropriateness of each learning method differs between intended application of the network. [9]

During the learning process, the weights are updated according to a predetermined optimization method. Generally, the optimization function is an algorithm that updates the network parameters to minimize a loss function. The choice of loss function and optimization function is done before the training starts.

The performance of the final network depends on how well the training has been executed. The two main factors that determine training conditions are the hyper parameters and the training data.

Hyper parameters

A hyper parameter is a parameter that is used to control the learning process in machine learning. When training a neural network, there are many hyper parameters that determines the structure and performance of that network.

One type of hyper parameter is the choice of loss function and optimization function. The loss function calculates how well the current weights perform on the data by comparing the output from the network with the ground truth. An optimizer uses the value from the loss function to update the weights in the network to minimize the loss.

In the deep learning application in Matlab, it is possible to choose from many implemented optimization functions, one of these is the Adam optimizer,[19]. When implementing the U-net structure using Matlab, a pixel classification layer is the last layer of the U-net. As a standard, this layer uses the cross-entropy loss function, [20] [21].

Training data

It is important to have a large data set available to ensure that the trained network has seen as many images as possible. The training data set should be large, but it should also have diversity so that many anatomical differences are included (for medical applications). It is also important that the training data set introduces different image qualities such as noise, contrast, and differences in brightness, when these differences are expected in the data that the network will be used on.

During supervised learning, where ground truth is included in the training process, it is important that the ground truth has been generated well. For networks trained for segmentation purposes, the ground truth segmentations should be generated carefully and with much attention to detail since the network will see this data as the true structure.

3.2 Medical Image Processing

3.2.1 Image Classification and Semantic Segmentation

One fundamental task in computer vision is image classification, where an image is given and an output is computed in the format of a single label, which can be seen in Figure 3.6. In image classification it is typically assumed that there is only one object in the image, and that the label represents that object. The task of semantic segmentation is more complex since the image is allowed to contain more than one object. The goal of semantic segmentation is to give each pixel in the image a label, corresponding to an object class. The output from a semantic segmentation task is a high resolution image where each pixel is labeled, see Figure 3.7. Note that image classification and semantic segmentation also can be done with other machine learning methods than neural networks.

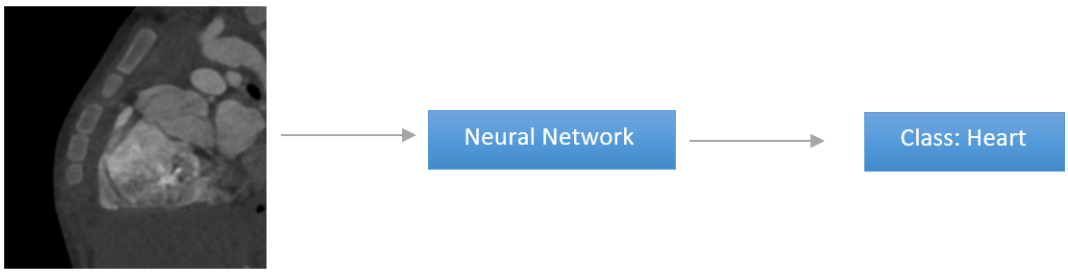


Figure 3.6: Image classification. Image by author.

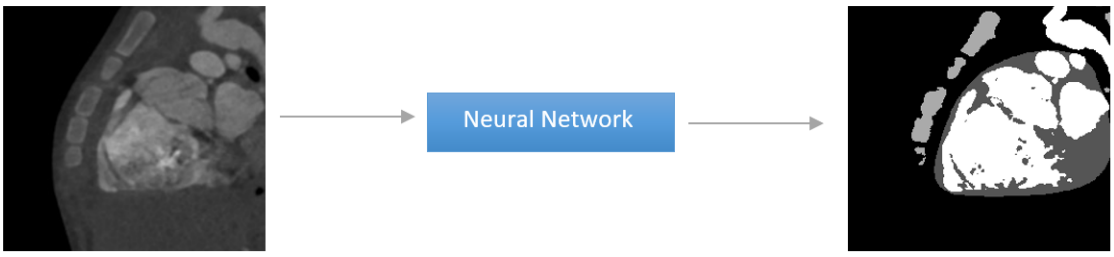


Figure 3.7: Semantic segmentation where each color represents a different class. Image by author.

3.2.2 Image Modality

In this project, the image modality is Computed Tomography (CT) images. The images are taken in a CT scanner with rotating x-rays and the resulting signals are then processed by the machine's computer to generate cross-sectional images, or "slices" of the body. Many slices can then be stacked together to form a 3D image of the patient, which makes the anatomy easier to study than in traditional 2D images. [22]

The images produced by a CT-scanner are grey-scaled images where high density structures, such as bone, are bright but low density structures are dark. Bones are therefore the easiest to identify and study. When other structures are of interest, a contrast agent can be injected to the patient to temporarily increase the brightness in that area. The contrast agent contains substances that are better at attenuating x-rays than the soft tissue. This can for example be an intravenous contrast agent that brightens the blood so that the heart can be studied. If the digestion system is of interest, it is more popular to use an oral contrast agent, [22].

3.3 Data Augmentation

When training a neural network, it is important to have a large data set available. In medical applications, it is a common problem that not enough data is available. Data augmentation is a method to increase the amount of data, without including more patients, where new data is created from already existing data by changing some properties.

Data augmentations can be done on whole images, or on smaller image patches. These patches are created from the training images to efficiently increase the amount of data available. The creation of these patches are determined by multiple hyper parameters which can be changed during training to optimize the performance of the network.

Data augmentations can include flipping or rotating the patches so that anatomy is visualized in different directions. It can also include changing the contrast or adding blurring to some images so that different image qualities are represented in the training data. How much flipping, rotating, blurring ect. that should take place is determined by augmentations parameters. The augmentation parameters are usually accompanied

by some probability factor that determines how often that specific augmentations should be performed.

3.4 Performance Evaluation

Evaluating the performance of image segmentation tasks is usually done by comparing the automatically created segmentation with the ground truth segmentation. In this project, two evaluation metrics where used. These are the Dice coefficient and the Jaccard index.

The Dice coefficient compares the automatic segmentation with the ground truth segmentation and produces a score of how similar these segmentations are. The Dice score is a measurement of how much the segmentations overlap between 0 and 1, but can be converted to a percentage. A score of 1 is achieved when the segmentations fully overlap. A visualization of this is presented in Figure 3.8.

$$Dice = \frac{2 \times \text{Area of Overlap}}{\text{Total Pixels Combined}}$$

The Jaccard index is a measurement of how many pixels are shared in the segmentations compared to distinct for each separate segmentation. The union is all classified pixels from both images, minus the overlap. It produces a score between 0 and 1, but can also be converted into a percentage, [23].

$$Jaccard = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

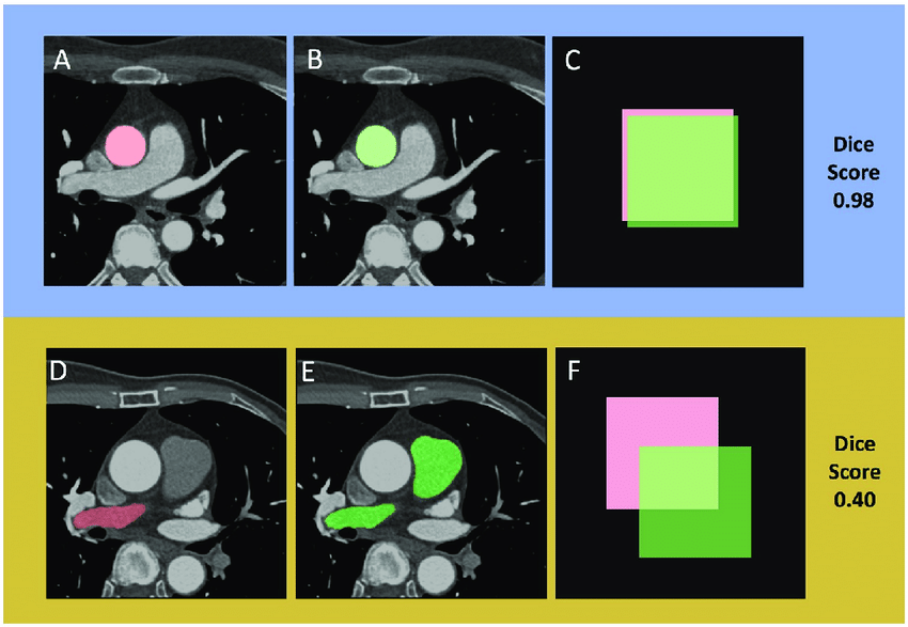


Figure 3.8: Dice score when two different segmentations are compared. In image A and B, two different segmentations have been done. As they are very similar, a high Dice score is computed. In image D and E, the two segmentations are not alike, and a low Dice score is computed [24]

4 Material

4.1 Data Acquisition

The material for this project was acquired through Skåne University hospital. Informed consent for the participants where waived by the Swedish Ethical Review Authority (Dnr 2021-03583). Images were completely anonymized retaining only age, gender and overall diagnosis. Images are taken locally at Lund University hospital with a Siemens CT-scanner of model type Somatom Definition Edge and Somatom Definition Flash.

To ensure that the algorithm became as robust as possible, a broad data set was used for training and testing regarding image and patient characteristics.

The purpose of this project was to create an algorithm that automatically segments children's hearts. It was therefor decided to exclude patients above the age of 18. The final data set used for training and testing included a spectrum of ages, ranging from 0 to 17. The sex was also taken into consideration, to ensure that the data set is a valid representation of the population.

Another important factor that was taken into consideration when creating the data set was the anatomy of the patients. To increase the robustness of the algorithm, a broad spectrum of diagnoses, illnesses and anatomical differences was included. Some differences in anatomy are interesting to include as that makes the network more robust, but some can negatively affect the performance of the network. The act of including appropriate data is a balance act, and in this project the decisions on which data to include and exclude was done together with physicians with great knowledge in the field.

Another advantage with including anatomy with the patient files was the fact that knowing the anatomy of a patient beforehand simplified the generating of the ground truth segmentations.

Regarding image characteristics, the data included images with a large spectrum of image contrasts, image intensities and image quality. It also included images taken with and without a contrast agent present.

4.2 Data

Table 4.1 shows the data used in this project. The anatomy of the patient, together with its age and gender is presented. This table includes both the training data and the test data.

Patient	Anatomy	Age	Sex
1	Partial Anomal Pulmunary Vein Oriface, VSD, PDA	1	M
2	Glenn Surgery	1.5	M
3	DORV, Pulmunary Stenosis	2	M
4	DORV, TGA, PS	8	M
5	Dextrocardia, TGA with PA	1.5	M
6	TGA	3.5	M
7	TCPC Surgery	3	F
8	Truncus type 1	0.5	F
9	Complicated Heart Defect, Glenn Surgery Awaits	0.2	M
10	VSD, Sinus Venosus Defect, PAPVD	0.2	F
11	DORV, TGA, VSD, PA	0.2	M
12	VSD	1	F
13	Aortic ring	14	M
14	TAPVR	0.4	M
15	Unknown	1	M
16	DORV	0.1	M
17	Double Aortic Arch	1	F
18	TGA, VSD, DORV	3d	M
19	PS, VSD, MAPCA	0.1	F
20	ASD, Pacemaker	13	F
21	PV anomaly	9	M
22	DORV	17	M
23	VSD/PAPVD, PDA	1	M
24	HLHS	1	M
25	PA, ASD	3.5	F
26	Situs Inversus	3	M
27	PA, VSD, MAPCA	15	F
28	HLHS	0,2	F

Table 4.1: Data. Anatomy refers to anatomical description or official diagnosis. The age is given in years, if nothing else is indicated (d is days).

5 Method

In this project, a neural network was implemented and trained to fit this particular problem. This was done on ground truth segmentations that the author herself created. The most time consuming part of this project was generating the ground truth segmentations, as this had to be done with great attention to details, and in a completely new software to the author of this thesis. Software training was performed by testing different tools and using these to segment five images. These segmentations were then evaluated by the supervisor, who is the director of the 3D Center and has 20+ years experience of medical imaging and image segmentation. The feedback from the supervisor was used to improve the author's method of creating segmentations.

Another major part of the work was to adjust hyper parameters to fit the actual task, as the network structure and segmentation pipeline was mostly in place already and had been successfully used in previous projects.

The performance of the network was evaluated after every change in hyper parameter or training data.

In the sections below, a more detailed description of the development of this project is presented.

5.1 Generating Ground Truth Segmentations

Manual delineations were used as ground truth in this project. This ground truth was generated using Segment 3DPrint and the output was MAT files. Since the process of creating manual delineations was a time-consuming process, it was done parallel to the rest of the project as more material became available.

Some delineations in this project were done in the beginning of the project to quickly get started with training the network. Only a few images were available at this time. The intentions was to go back to these initial delineations and redo them when a good system for training had been set in place, which was done a few months into the project when the author had improved the method of delineating the data, compared to the beginning of the project.

The delineations were done with the highest possible accuracy to ensure the input to the network would be accurate. This was done by using appropriate tools in Segment 3DPrint. This meant making sure that all chambers where correctly separated when appropriate, that all vessels where included and that all tissue had been classified as the correct class.

5.1.1 Visualization in Segment 3DPrint

In Segment 3DPrint, visualization of the CT images are given in the transversal, sagittal and coronal view plane at the same time, see Figure 5.1. The delineations can be visualized with a thin green line, representing the border of the object, or as a red color that fills the entire object. An example of a segmented heart is presented in Figure 5.2. The images can be stacked together to create a 3D volume, which can be visualized in the software, see Figure 5.3. In the software, it is possible to interact with the 3D volume.

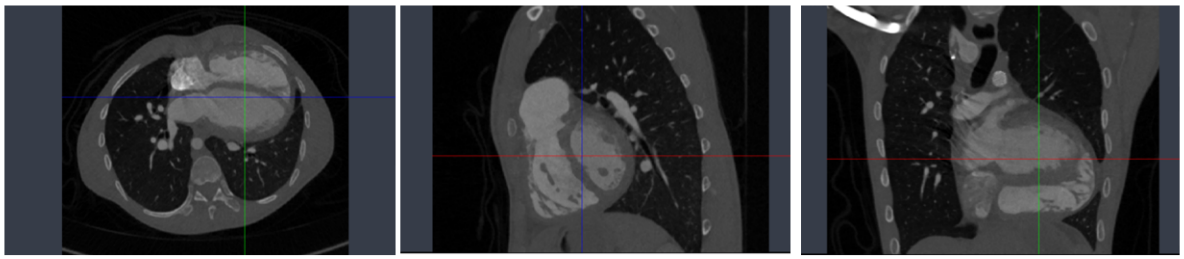


Figure 5.1: View of CT images in Segment 3DPrint. From left to right is the transversal, sagittal and coronal view.

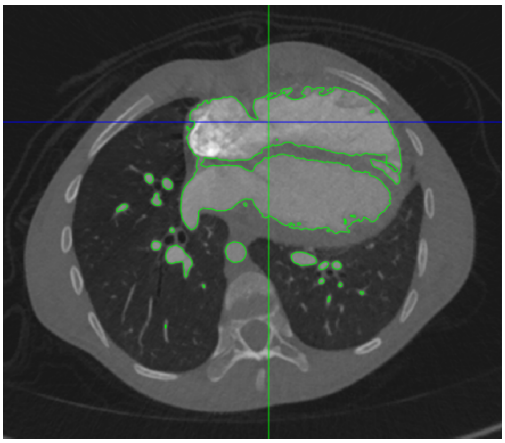


Figure 5.2: Transversal view of a heart with manual delineations represented with a green contour

Figure 5.3: 3D view of a manually segmented heart

5.1.2 Segmentation Classes

When performing segmentations tasks, one or more classes need to be defined. During segmentation, each pixel will be given a label corresponding to one of these classes. The most central class in the project was the Blood Pool, which is the cavities of the heart and the vessels, see Figure 5.3. In most CT images, the bones is the brightest tissue, not the Blood Pool (apart from when a contrast agent is present). This meant that the bone structure was the most easily detectable structure in the images, and therefor the most easy to segment. During the project, it was hypothesised that the network would perform better on the Blood Pool segmentation if a second class (a bone class) was also included, see Figure 5.4. This hypotheses was based on the concept of multi-task learning in machine learning, which says that if there is some correlation between tasks, the learning process is improved when you include multiple learning tasks.

In this project, that meant that the inclusion of multiple classes should improve the networks ability to find underlying principles or information that is shared between classes. This hypothesis was tested by including and excluding the Bone class in training, and do performance evaluation on the Blood Pool segmentation.

Based on the same hypothesis, a third class was included, this is a class called Bright. In this class, foreign objects such as catheters, electrodes and shunts were included. These types of objects are very bright in CT images since they often are made of high density materials.

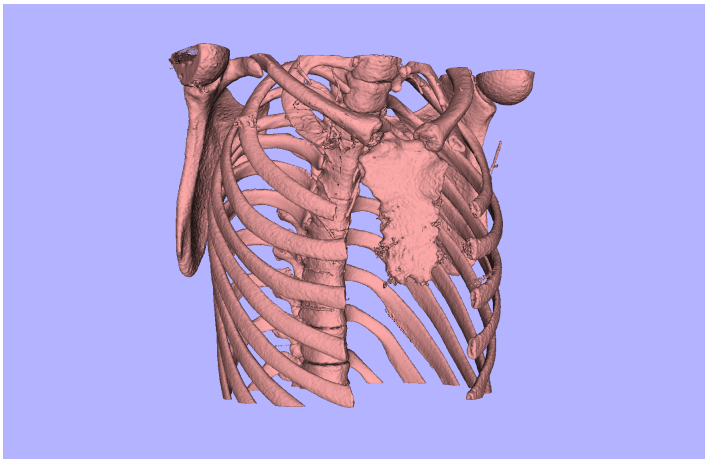


Figure 5.4: 3D view of manually segmented bones

Another class was created called the Pericardium, which includes the pericardium. This class was included since the pericardium is often included in the 3D models of the heart structure, and is therefore of interest to those who creates and views these models. See Figure 5.5, to see how this class looks together with the Blood Pool in 3D.

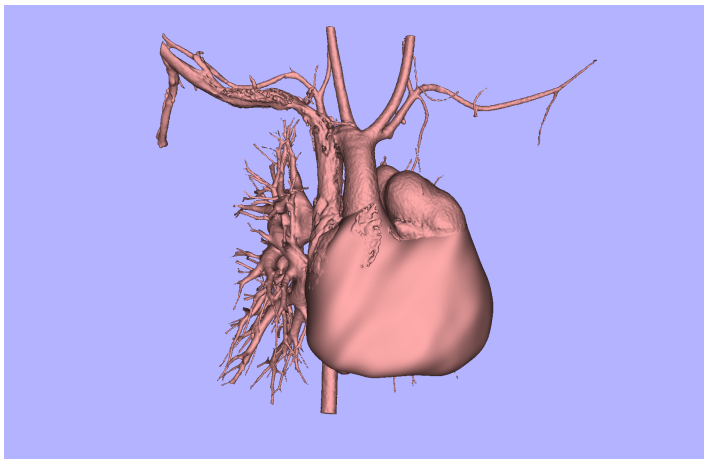


Figure 5.5: 3D view of manually segmented Pericardium together the Blood Pool.

5.1.3 Manual Segmentation

The initial segmentation on an image was done by setting an image intensity threshold and classify everything below that threshold as Background. The Background class is not included in the manual segmentation but rather as one of the parameters in the network structure. The class is created automatically by the network when it views the training data. Every pixel that hadn't been classified in the training data was defined as Background.

The pixels above that threshold were then separated, and classified as either Blood Pool, Bone or Bright by manual outlining. This was done with different tools in the software (Segment 3DPrint), such as dividing, smoothing and filling tools. The Pericardium class was created by manually outlining the outer rim of the pericardium in a few image planes and then interpolate over all planes to create a solid body.

Image intensity and contrast differed for all patients. This meant that the threshold intensity for segmentation had to be adapted for every patient, making it impossible to create a one-size-fits all solution for segmentation. The image intensity was chosen by visually determining an appropriate threshold for each patient that would include the most central anatomy.

For every patient, a file containing the segmented Blood Pool, Bone, Bright and Pericardium was created and exported from Segment 3DPrint. These segmentations were then used as ground truth while training the network and as test data to evaluate the performance of the network after training.

The delineations were done by only one person to make sure they were as similar as possible. While creating the ground truth segmentations for this project, the delineations were done by the author of this project, since her knowledge about the physiology of the heart was deemed adequate. To make sure the delineations were correct, the work was continuously evaluated by people with more knowledge in the field.

As more data became available to the author, more delineations could be done and

the training data set could be expanded. This also meant adjusting how the data set was divided into training and test.

5.1.4 Hierarchy within Class Labels

In this project the class labels were Blood Pool, Bone, Bright, Pericardium and Background. There is a hierarchy within these labels that is defined in the network structure. This hierarchy is needed when the training is taking place and the network analyzes the ground truth segmentations. It could be that the ground truth segmentation is done so that one pixel is assigned multiple labels, for example a pixel between the Blood Pool and the Pericardium is assigned both labels. By introducing a hierarchy, the network learns that which label it should prioritize, if there are many to choose from. In this project, the hierarchy was defined as followed (with the most important first): Blood Pool, Bone, Bright, Pericardium and Background.

5.2 Data Split

In the beginning of the project, when only a small amount of data was available, a random split of training and test was done. Towards the end of the project, where more data became available, a more conscious split could be done. This conscious split was done so that a good representation of anatomy was included in the training data.

The test data set contains anatomies that were not included in the training set so that the network can be tested on structures that it has never seen before. In this project, a heart with calcium depositions is included in the test data, but not in the training data. The reason for not including any patients with calcium depositions in the training data was that not enough patients with this anatomy was available.

In 5.2, the final training data is presented, and in 5.1, the final test data is presented.

Patient	Anatomy	Age	Sex
22	DORV	17	M
23	VSD/PAPVD, PDA	1	M
24	HLHS	1	M
25	PA, ASD	3.5	F
26	Situs inversus	3	M
27	PA, VSD, MAPCA	15	F
28	HLHS	0,2	F

Table 5.1: Test data. Anatomy refers to anatomical description. The age is given in years, if nothing else is indicated (d is days).

Patient	Anatomy	Age	Sex
1	Partial Anomal Pulmunary Vein Oriface, VSD, PDA	1	M
2	Glenn Surgery	1.5	M
3	DORV, Pulmunary Stenosis	2	M
4	DORV, TGA, PS	8	M
5	Dextrocardia, TGA with PA	1.5	M
6	TGA	3.5	M
7	TCPC Surgery	3	F
8	Truncus type 1	0.5	F
9	Complicated Heart Defect, Glenn Surgery Awaits	0.2	M
10	VSD, Sinus Venosus Defect, PAPVD	0.2	F
11	DORV, TGA, VSD, PA	0.2	M
12	VSD	1	F
13	Aortic ring	14	U
14	TAPVR	0.4	M
15	Unknown	1	U
16	DORV	0.1	M
17	Double Aortic Arch	1	F
18	TGA, VSD, DORV	3d	M
19	PS, VSD, MAPCA	0.1	U
20	ASD, Pacemaker	13	U
21	PV anomaly	9	M

Table 5.2: Training data. Anatomy refers to anatomical description. The age is given in years, if nothing else is indicated (d is days).

5.3 Performance Evaluation

The performance of the network was evaluated by looking at the automatic segmentation of the Blood Pool from each network, and then compare this segmentation to the ground truth segmentation. This was done with the Dice score and Jaccard index. These metrics were computed for all networks and then compared to each other to evaluate the performance on the test data. Since the main purpose of the network is to automatically segment the Blood Pool, this class was the only one included in the performance evaluation. By doing this evaluation after every network was created, it was possible to see if the latest changes in the network led to better performance or worse.

The Dice score and Jaccard index is a good measurement of how well the segmentations overlap, it is however important not to blindly rely on these metrics for evaluation. The segmentations also had to be visually investigated to evaluate the performance. The score together with visual evaluation was proven to be the best method, since a good score not always indicated a good segmentation.

The ground truth segmentations in question was the manual delineations created by the author of this report. It is well known that manual delineations are not created exactly the same every time, even if the delineations are done by the same person every

time. This is because every delineation includes many small decisions, and by making different small decisions, the outcome can vary. To evaluate just how much difference there were between the delineations created by the same person, the author delineated the same images multiple times during the project and compared these segmentations. This meant that a new delineation was compared to older delineations created by the same person, but created a few weeks later. This is important to include since the "intra-observer" difference can be put in relation to the "intra-network" comparison.

5.4 Training the Network

During training of the network, hyper parameters and data augmentations parameters were changed to see how they affected the performance of the network. These changes were done systematically, by only changing one parameter at the time, to be able to evaluate the effect of the change.

5.4.1 Hyper Parameters

Some hyper parameters were kept fixed throughout the project. This was decided since it wasn't possible to test all parameter combinations due to lack of time. The fixed parameters were chosen after studying the original U-net structure [17], and other projects. The fixed parameters and the ones that are adjusted during training, are presented in 6.3. The final parameter values are presented in the results section.

Fixed hyper parameters	Adjusted hyper parameters
Max Epochs	Mini Batch Size
Initial Learning Rate	Patch Size
Learning Rate Drop Period	Encoder Depth
Learning Rate Drop Factor	
Empty Patch Probability	
Minimal Patch Content	
Black Patch Probability	

Table 5.3: Hyper parameters. The left column gives parameters that are kept fixed to their default values during training. The right column gives the parameters that are adjusted during training.

The maximum number of epochs during training is determined by Max Epochs. One epoch is defined as the full pass of the training algorithm over the entire training set [25].

As mentioned in 3.1.4, the ADAM optimizer is chosen for this project. Connected to this optimizer are multiple hyper parameters that needs to be determined. These are the Initial Learning Rate (learning rate at the start of the training), Learning Rate Drop Period (how often the learning rate is decreased in terms of epochs) and Learning Rate Drop Factor (how much the learning rate is decreased with).

Mini Batch Size is the parameter that determines the size of the mini-batch of training data that is used to evaluate the gradient of the loss function and update the weights. It is desirable to have as high value as possible here since this parameter determines how many images are included when the weight of the network is update, which is an important step in the training of the network.

In this project, the training is done on smaller patches of each image. The creation of these patches is controlled by the following hyper parameters, the Empty Patch Probability (the probability of creating the patch even though it is empty), Minimal Patch Content (how much the patch must contain to be written), Black Patch Probability (probability of creating the patch even though it is black) and Patch Size. Each of these parameters were kept fixed to their defaults during training except for Patch Size.

The depth of the encoding part of the U-net is determined by the parameter Encoder Depth. This parameter determines how many encoding steps that will be done in the network. Increasing this parameter is generally a good practice, as it increases the number of times an input image is downsampled. More downsampling generally leads to better performance of the network since more features can be detected if lower resolution objects are included. However, increasing the encoder depth also increases the size of the network.

5.4.2 Data Augmentation

Some augmentation parameters were kept fixed throughout the project but some were changed to see how they affected the performance. All data augmentation parameters are presented in 6.5.

Fixed augmentation parameters	Adjusted augmentation parameters
Rotation	Contrast
Flip left/right	Flip up/down
Transpose	Translate
Brightness	Scaling in x-direction
Gaussian Blurring	Scaling in y-direction
	White Noise
	Streak Noise

Table 5.4: Augmentation parameters. The left column gives parameters that are kept fixed to their default values during training. The right column gives the parameters that are adjusted during training.

The probability of flipping the image up/down was changed during training. The reasoning behind this change was that there is a low probability that flipped images of this manner will ever be used as input to this network in practice (CT images of the thorax is usually in the upright position).

Scaling the image patches in the x- and y-direction was done by multiplying the size of the image with a factor that is randomly taken from a distribution centered around 1. The size of this distribution was changed to evaluate how that affected performance.

The probability of adding noise to the patches was changed, both for white noise and streak noise. Also, how much noise that was added was adjusted in terms of percentage of intensity range of the image patch.

Finally, the parameter determining how much contrast that should be added to the patches was changed. The probability of the augmentation to happen was kept the same, but the distribution of the contrast change was adjusted.

5.4.3 Multi-task Segmentation

Multi-task segmentations means that multiple segmentation classes are included in the training, with the thought that the learning process of the network is improved by this.

The most central classes in this project is the Blood Pool together with the Pericardium. The inclusion of the classes Bone and Bright is done to improve the automatic segmentations of the Blood Pool and Pericardium. The performance of the network was evaluated when including or excluding these classes, to determine if that was the case.

5.5 Network Output

The output from the network is segmented images where the Blood Pool or the Pericardium has been segmented. These are given in seperate images. In order for the segmentation to be clinically useful in some specific applications, another segmentation is created. This is called the Cardiac Model, and is a combination of the Blood Pool and the Pericardium, with some alterations.

The Cardiac Model is not automatically created by the network, but rather hard coded in the segmentation software using the network output as input data. This is done by using already implemented segmentation methods. Firstly, the Blood Pool is made hollow, with a shell outside to simulate the vessel wall. The hollow Blood Pool is then joined together with the Pericardium. A function called Close is applied, which closes small holes in the segmentations, and then a function called dilate is applied, which dilates the segmentations with few pixels. A smoothing function is then applied to smooth the edges of the segmentation. This combination of alterations creates the final Cardiac Model. In this, all volume in the pericardium that is not occupied with Blood Pool is filled to indicate myocardial tissue.

The combination of alterations that presented the best Cardiac Model was found after testing multiple combinations and visually evaluating the models.

5.6 Implementation

The software used in the hospital today is called Segment 3DPrint. The author of this report worked closely to the provider of this software [7], which made it possible to include the resulting method in the software. The automatic segmentation method proposed in this thesis was implemented and incorporated into the software.

The function is up and running on a beta version of the software, and will be included in the next clinical release of the software.

The function is used by uploading CT images into the software, finding the menu that includes all automatic functions and choose either AutoBloodPool or AutoCardiac-Model. The AutoBloodPool function creates a segmentation of the Blood Pool and the AutoCardiacModel function creates the Cardiac Model that is hollow and includes the pericardium.

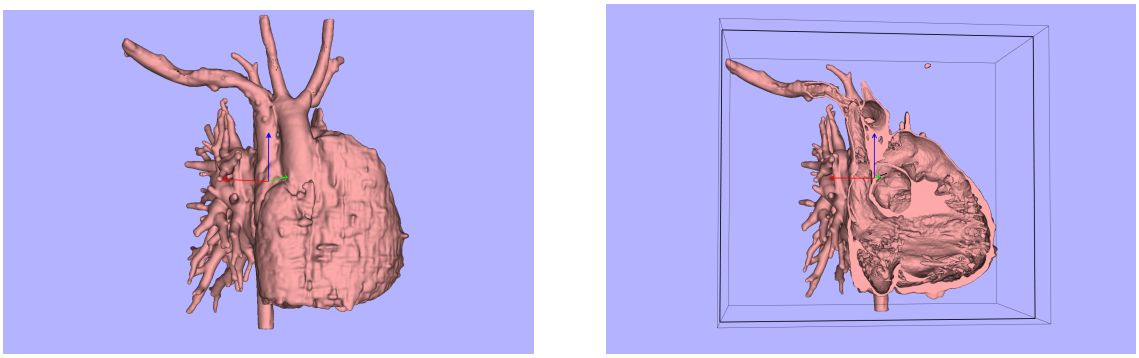
6 Results

In this section, the final output from the network will be presented visually in the format of the Cardiac Model, which is a combination of the Blood Pool segmentation and the Pericardium segmentation. For the final network, Dice score and Jaccard index have been computed on the test data, and is presented. The Dice score and Jaccard index have also been computed on the intra-observer segmentations, and is presented below.

A visual representation of segmentations with high and low dice scores will be given as well as the resulting network parameters.

6.1 Performance

The final output in the format of the Cardiac Model where the Blood Pool and Pericardium segmentations has been joined together. In Figure 6.1, a view of the segmentation is given in 3D and in Figure 6.2, the same segmentation is given in 2D. This heart belongs to patient 22.



(a) The Cardiac Model **(b)** Cross section of the Cardiac Model

Figure 6.1: The final segmentation of the Cardiac Model. This is the class Blood Pool and Pericardium together. The Blood Pool is hollow to represent a life-like model.

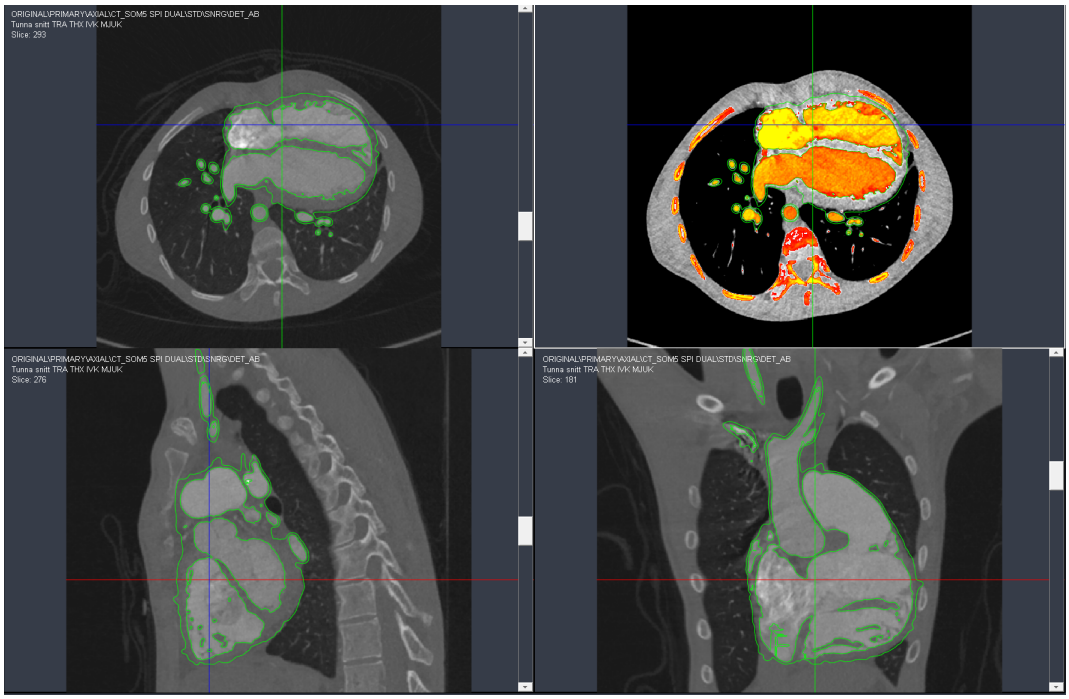


Figure 6.2: The final segmentation in the transversal (top left), sagittal (bottom left) and coronal (bottom right) view. This is the Blood Pool and the Pericardium classes together, as they form the Cardiac Model. The image in the top right is the transversal view with a simple threshold segmentation on it, this view is not of importance when studying the final output, but rather used as a help to segment images from scratch.

The Dice score and Jaccard index have been calculated to evaluate the performance of the network. These metrics are calculated on the test data by computing the scores between the ground truth segmentations and the automatically created segmentation. In 6.1, the scores are given for each patient in the test data and from that, the mean value and standard deviation is calculated and presented.

Patient	Dice Score (%)	Jaccard Index (%)
22	96,5	93,3
23	92,7	86,3
24	93,9	88,9
25	95,6	91,5
26	94,1	88,8
27	86,9	76,9
28	95,0	90.5
Mean ± SD	93,5 ± 2,9	88,0 ± 5,0

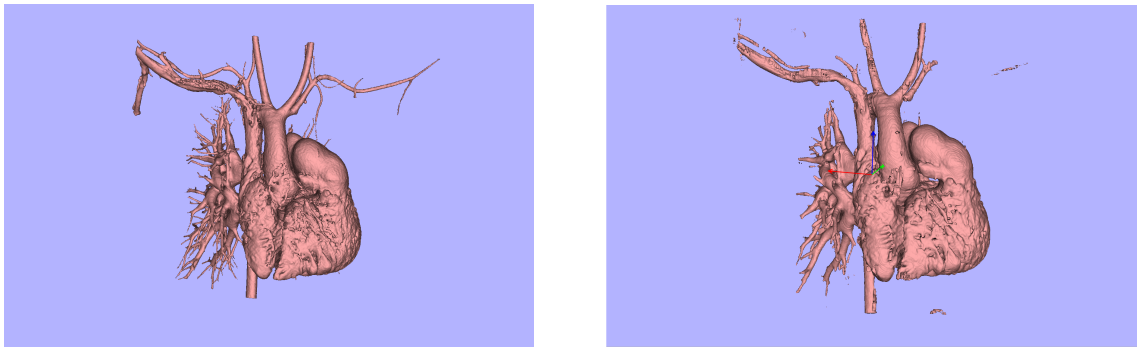
Table 6.1: Resulting performance on test data. Dice score and Jaccard index have been calculated between the ground truth segmentations and the automatically created segmentations

The intra-observer segmentations have been evaluated and the metrics are presented in 6.2. This comparison is done between ground truth segmentations done on different occasions by the same person, which is the author of this thesis. Two different data sets were used for this evaluation.

Patient	Dice Score (%)	Jaccard Index (%)	Interval (Days)
2	99,4	98,8	7
2	98,1	96,2	8
2	95,4	91,2	35
3	99,0	98,1	7
3	96,4	93,1	8
3	96,4	93,0	35
Mean ± SD	97,5 ± 1,5	95,1 ± 2,8	

Table 6.2: Resulting performance of intra-observer evaluation. Dice score and Jaccard index have been calculated between segmentations done by the author on different occasions. The time between these occasions is presented in the fourth column.

In Figure 6.3, two segmentations of the Blood Pool are shown. These are from the same patient, one is the ground truth segmentation and the other one is the automatically created segmentation. This is an example of a good segmentation in the test data, the Dice score between these segmentations is 96,5%. In Figure 6.4, the difference between the segmentations is presented. Here, the areas which have been segmented differently are visualized in a green colour. This figure also includes a 3D visualization of these differences. This heart belongs to patient 22.



(a) Ground truth segmentation segmentation **(b)** Automatically created segmentation

Figure 6.3: Automatic segmentation of the heart. This is the class Blood Pool.

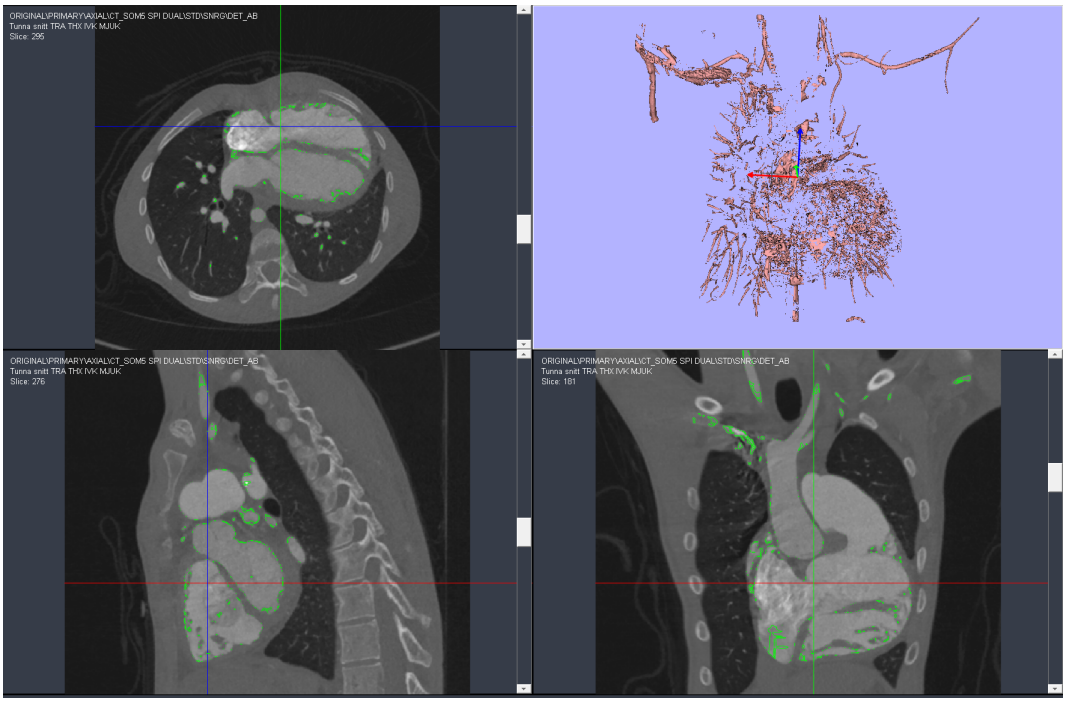


Figure 6.4: Differences between ground truth segmentation and the automatically created segmentation from the network. The green areas are pixels that the network segmented differently compared to the ground truth. The 3D image in the top right corner is a 3D visualization of this green area.

In Figure 6.5, an example of a relatively bad segmentation of the Blood Pool is given. The Dice score between these segmentations is 86,9%. In 6.5, the areas which have been segmented differently are visualized in a red colour. This figure also includes a 3D visualization of these differences. This heart belongs to patient 27.

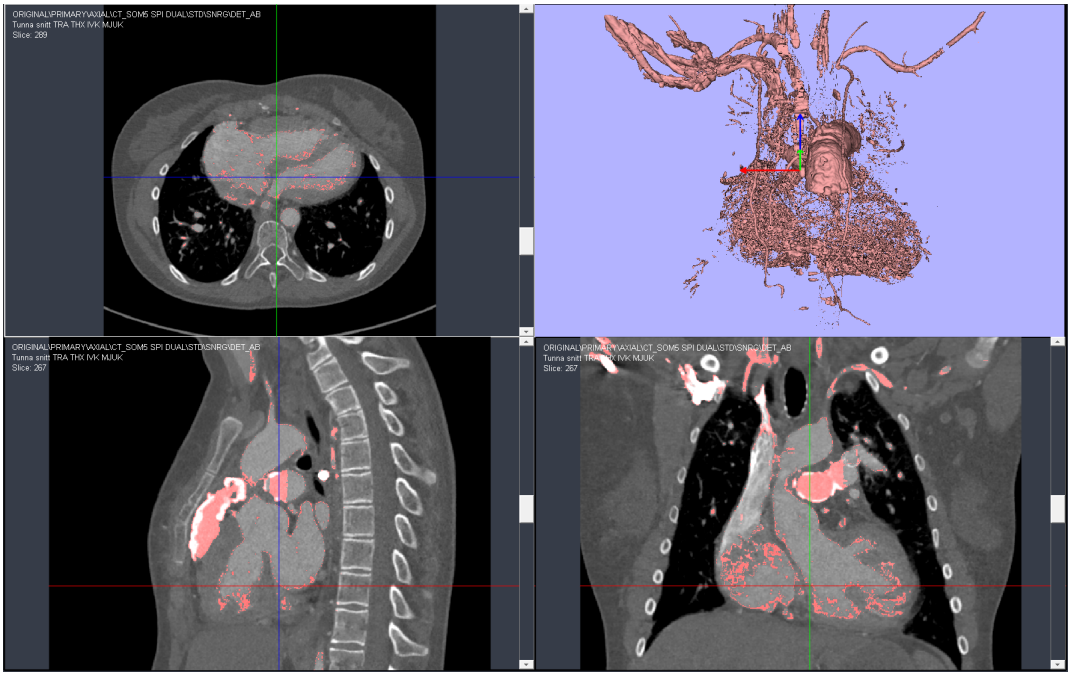


Figure 6.5: Differences in segmentation between ground truth segmentation and the automatically created segmentation from the network. The red areas are pixels that the network segmented differently compared to the ground truth. The 3D image in the top right corner is a 3D visualization of this red area.

6.2 Network Parameters

6.2.1 Hyper Parameters

The default value of the parameter Mini Batch Size in Matlab was 30, so this was the initial value. This parameter value should be maximized to improve the learning process during training. Trying to run a training iteration with this initial value did not work because of limitations of the GPU. The value was decreased one step between each attempt so that the highest value that the GPU could handle could be found. This was found to be 20.

The Patch Size determines how big the patches should be. It was initially set to 128, as this was the default value in the used segmentation framework. This meant that the patches that were created were 128 pixels in size. This parameter was later increased to 256 pixels, which was shown to increase the performance.

The default value of Encoder Depth in Matlab is 4. Since a deeper network was thought to increase performance, this value was changed to 5 during training. As this

was proven successful, this was the final value. No training was done with a value higher than 5.

The final hyper parameter values are presented in 6.3 and 6.4.

Training parameter	Value
Max Epochs	100
Initial Learning Rate	1e-3
Learning Rate Drop Period	4
Learning Rate Drop Factor	0.8
Empty Patch Probability	0.01
Minimal Patch Content	0.025
Black Patch Probability	0.005

Table 6.3: Fixed hyper parameters.

Training parameter	Value
Mini Batch Size	20
Patch Size	256
Encoder Depth	5

Table 6.4: Adjusted hyper parameters.

6.2.2 Data Augmentation

The probability of flipping the image up/down was changed to 0% during one training to evaluate if that improved the performance. Changing this parameter to 0% did not increase performance, but rather lead to a slight decrease in performance. This meant that the final value was set to its initial value of 50%.

The initial probability of translating the image patches was 75%. This was lowered to 50% and since an increase in performance was noted, this was adjustment was kept.

Scaling the image patches in the x- and y-direction was done on 100% of the images according to a distribution centered around 1. The size of this distribution was originally set to U(0.7, 1.3) for both directions, but changed to a broader spectrum to see if the change would affect the performance. As the performance increased after this effect, the broader distribution of U(0.6,1.4) was chosen.

The probability of adding noise to the patches was changed, both for white noise and streak noise. It was changed from 10% to 20%. The distribution was adjusted for the streak noise from U(0,70%) to U(0,100%) and for the white noise from U(0,20%) to U(0,15%).

The probability of the contrast augmentation to happen was kept the same, but the distribution of than augmentation was changed. This was changed between U(0.8, 1.2) and U(0.7,1.3).

The final augmentation parameter values are presented in 6.5 and 6.6.

Augmentation	Probability	Distribution	Unit
Rotation	25%	U(-45,45)	Degrees
Flip left/right	50%	-	-
Transpose	50%	-	-
Brightness	25%	U(-100,100)	HU
Gaussian Blurring	25%	{3,5,7}	Pixels (Of filter size)

Table 6.5: Fixed augmentation parameters. The left column gives the method of augmentation and the second is the probability of that augmentation to occur. The third and fourth column gives the distribution of the augmentation parameter and its unit.

Augmentation	Probability	Distribution	Unit
Flip up/down	50 %	-	-
Translate	50%	-	-
Scaling in x-direction	100%	U(0.6,1.4)	Factor
Scaling in y-direction	100%	U(0.6,1.4)	Factor
White Noise	20%	U(0,15%)	Of intensity range
Streak Noise	20%	U(0,100%)	Of intensity range
Contrast	25%	U(0.7, 1.3)	Factor

Table 6.6: Adjusted augmentation parameters. The left column gives the method of augmentation and the second is the probability of that augmentation to occur. The third and fourth column gives the distribution of the augmentation parameter and its unit.

6.2.3 Multi-task Segmentation

Including the Bone class was thought to improve the performance of the network, and according to visual evaluations of the segmentations and the evaluation metric comparison, it was shown that it did actually improve the performance. Including the class Bright was also thought to improve the performance, but it did not. This was concluded by visually studying the automatic segmentation, where it was obvious that the inclusion of the class Bright led to worse segmentation of the class Blood Pool.

7 Discussion

In summary, a method for automatic 3D segmentation of hearts with congenital heart defect was developed. Ground truth segmentation were created by the author, and then used to train the network. In the Results section, a comparison between the ground truth segmentation and the automatic segmentation for images with a high and low score, was given to the reader.

The evaluation metrics were given for all test images, and the mean Dice score of these images was 93,5% (±2,9) and the Jaccard index was 88,0% (±5,0). The intra-observer Dice score was on average 97,5% (±1,5) and the Jaccard index was 95,1% (±2,8).

Many choices were made during training, like hyper parameter choices and data augmentations, to adjust the network to the task and the final values of these parameters were presented in the Results section.

7.1 Generating Ground Truth Segmentations

Generating the ground truth segmentations was done by one person. The reason for this was to ensure that the delineations were consistent, since everyone segments images differently. A disadvantage with this is that it created room for possible errors, even though the segmentations were evaluated by other people. It might be that the author segmented something incorrectly that was not found during the quality assurance process. However, having multiple people to delineate the images would probably introduce more bias, than having one person do it.

Preferable, the ground truth should be generated by someone with in depth knowledge in anatomy and congenital heart defects so that the ground truth segmentations are generated more thoroughly. An alternative to this would be to let the segmentations done by the author go through a more thorough quality assurance process.

7.2 Evaluation Method

The evaluation metrics chosen for this project was the Dice score and Jaccard index. These were chosen because they are standard metrics for semantic segmentation tasks. A choice was done to focus on testing different combinations of hyper parameters and data augmentation to improve the network, and not evaluating different metrics parallel to this. Comparing different evaluation methods could be beneficial to asses if others would fit this task better.

The combination of studying the evaluation metrics and visually examining the auto-

matic segmentations turned out to be a good method. In some cases, a segmentations was given a high Dice score and Jaccard index, but a visual evaluation showed that the segmentation completely missed important anatomy.

Quantification of the visual evaluations would have improved the evaluation method in this project. In this project, only the author visually determined if the automatic segmentations were of good quality. Having a method in place for quantifying the performance of the automatic segmentations visually, would have made it possible to easier compare performance. For example, a system could be put in place where each segmentation is rated on a scale of 1 to 5, where 1 is a bad segmentation that need to be redone completely to be able to use it clinically, and 5 is a perfect segmentations that can be used clinically instantly. By having a rating system like this in place, it would also have made it possible to include other people to evaluate the segmentations. In that way, physicians and engineers that use these segmentations on a daily basis could have a say in which segmentations that are useful or not. Generally, it is very hard to quantify visual assessment, but including a scale with very clear definitions of the different steps, would have made it easier.

In this project, performance evaluation was done on the Blood Pool class, not the Pericardium class, even though both of these classes are important to the end result. The reason for this was that using the Pericardium class as an evaluation class, together with the Blood Pool class, was proven difficult. The manual segmentations of the Pericardium were relatively different to the automatically created segmentations, and when the evaluation metrics were computed, the scores did not accurately match the resulting segmentations. By visually evaluating the automatic segmentations, it showed that there were seldom a correlation between a high score and a good segmentation, visually speaking. It was then decided that the Blood Pool was going to be the only class used for evaluation. The Pericardium was then visually evaluated for each network, to see how well the segmentations were created.

7.3 Performance

The performance of the network is presented in the format of a mean Dice Score and mean Jaccard Score. These scores tells us how much alike the automatic segmentations is to the ground truth segmentation. For this network, the average Dice score is at 93,5%. When this automatic method is used clinically, this means that an automatic segmentation is created that almost completely matches a segmentation that the users usually need to create manually. A manual segmentation can take 30-90 minutes, according to users that the author talked to. The resulting segmentations can be visually examined by the user to evaluate how well it has been generated, and if the user find errors that they wish to fix, that is possible in the segmentation software. This means that instead of manually outlining the heart, the user can generate an automatic segmentation and then make small changes if needed, a process that might take 15 minutes.

In the results section, two examples are given of Blood Pool segmentations, see Figure 6.4 and Figure 6.5. In both of these images, a 3D visualisation of areas that have

been classified incorrectly by the network is given. In Figure 6.4, it is noticeable that no large structures have been missed, only some smaller vessels. These types of errors are not severe in clinical applications. The reason for this is that almost all cases of 3D segmentation in clinical usage, are done with interests in the large structures, not the small vessels. This means that the segmentations in 6.4, could possibly be used immediately. However, in Figure 6.5, some large structures are missed by the network. Most of the Blood Pool have been segmented correctly, but one large vessel and many smaller vessels are missed. When the end user visually evaluates the automatic segmentation, they would easily realize that this vessel is missing. The segmentation software that has implemented this method includes many efficient tools to correct this mistake. A "bad" segmentations is therefore not necessarily useless.

The aim of the method is not to completely replace manual segmentation, but to make it faster and easier to create 3D models by providing an automatically created segmentation that only needs adjusting. The fact that this network performs well, means that the automatic segmentations are good enough so that those who usually creates the segmentations manually, will save time by only needing to make adjustments, which is a great gain.

As mentioned earlier, it is interesting to put the performance of the network in relation to the intra-observer comparison. This shows that if one person would manually segment the same image over and over again, the segmentations would only be 97,5% similar according to the Dice score. One interpretations of this is that the network will not have to outperform this to be considered good, since no human will be able to outperform it.

7.4 Data

The data set for this thesis was provided to the author over the course of the entire project. In the beginning, only a small amount of the data was available. As a result, the ground truth segmentations were done during different stages of the project. As seen in the intra-observer results, there is a larger difference between segmentations when the time between the segmentations is longer. This means that the ground truth segmentations could have been created differently, depending on if they were done in the beginning of the project or in the end. In this project, the interval between segmentations is as much as three months. The fact that there could be unwanted differences in the ground truth data means that the network training could have been negatively affected. This is because the network could have been provided with contradicting information regarding the segmentations, that makes it hard for the network to find patterns. Note that some segmentations were done in the absolute beginning of the project to learn the software, these segmentations are not included in the training or test data.

When training a neural network the training data set has an important impact on the performance of the network. In this project, many different types of diagnoses and anatomical differences were included, which is good for the robustness of the network. Testing the network on an image with anatomies that the network hadn't

previously seen is an important method for evaluating how well the network performs. In the results section, a segmented heart with calcium depositions is shown in 6.5. The training data set did not include any patients with calcium depositions, which most likely contributed to the relatively bad segmentation of this Blood Pool. This shows that the network is not as robust as is could be. If the network should be able to handle new anatomy well, the training data set needs to be extended by multiple patients with other anatomies, like calcium depositions. But something that is worth noting is the fact that the diagnosis HLHS was also only present in the test data, not the training data. The network did however perform well on this "unseen" anatomy, compared to the calcium depositions. Finding which anatomies or diagnoses that the network would benefit from including in the training data, is probably best done by consulting physicians who work with these patients on a daily basis.

In this project, the network is applied to a set of patients with many different anatomies and diagnoses. This means that the network is able to handle multiple anatomies well. The fact that the network is able to segment a broad spectrum of anatomies is an advantage in most cases, since it can be applied in many situations. However, it is possible to create a network that specializes in one type of anatomy or diagnose. This would happen if the training data only included one certain anatomy. This would probably lead to that network performing better on that specific anatomy, compared to how the network in this project would perform on that specific anatomy. Creating specialized networks require a big data set containing a specific anatomy, which is hard to come by since the resources in most cases are limited. This is especially true regarding medical data. Trying to find enough data on all the anatomies represented in this thesis, in an effort to create multiple specialized network, would be very hard since some anatomies are more rare than others. A specialized network could be of interest if one type of anatomy is over-represented in the patients, but the gain of having a relatively generalized network outweighs specialized networks because of how useful they are clinically and because finding enough data to train it is easier.

The test data set is not as large as desired. More test data would be interesting to include so that the performance of the network could be evaluated better. The reason for the small test data set is that manual segmentation of the test images had to be done in order to include images in the data set, and this is a time consuming process. With more time, more test data could be segmented and a better performance evaluation could be done.

A common problem with deep learning methods is their ability the be generalized. Trained networks generally don't perform well on unseen data that differs too much from what is was trained on, this can be data from another scanner type, another image modality, or another hospital (since image taking routines can differ). This is a limitation that makes it harder to implement deep learning solutions to more general problems. One method of improving the generalization is the patch based solution where image patches are augmented to introduce differences in training, which was used in the project.

In this project, a choice mas made to focus on images that have been produces by the same scanner, on the same hospital and visualized approximately the same anatomy (since we focus on the thorax). This means that the automatic segmentation method

in this project might not be applicable on other anatomies, other hospitals, or other scanner, without some tweaking, since patch based solutions isn't enough. The generalization could be further improved by including images of different types, for example images taken on different scanners.

7.5 Network Parameters

7.5.1 Hyper Parameters

The Patch Size was changed from 128 pixels to 256 during training. This change occurred after the network with value 128 was evaluated on test images. It was then obvious that the automatically created segmentations weren't performing well in some areas. This was thought to be a result of the small value of Patch size since the network was trained on smaller areas of the image thus lacking an overview of the anatomy. By increasing this value to 256, it was expected to increase performance in this matter, which it also did.

Increasing Encoder Depth from 4 to 5 was proven to improve the performance of the network, which was expected. This is because with an increased depth of a U-net, images are downsampled more. This allows for more low-resolution features to be detected than in higher-resolution images. More features generally leads to a better learning process, which results in a network that performs better. The reason for not increasing the depth to 6 is that the size of the network with Encoder Depth 5 was quite big. And since the network is to be included in a software, it was preferable to not increase the size more than that.

The choice of loss function and optimization method for training was not done by the author of this thesis. These were chosen because of the already implemented network structure provided included these settings. Since they had proven to perform well on similar tasks before, it was decided that they would be used in this project. It is however possible to replace these if desirable. It would have been interesting to see how changing these affected the learning process and the performance of the network.

Many different combinations of hyper parameter values were tested in this project. However, not all combinations were tested because of how difficult it is to find all possible combination and because of a lack of time. This means that there might exist a better parameter combination for this task, that was not found. As an improvement to this project, more parameter combinations could be tested, as well as other loss functions and optimization methods.

7.5.2 Data Augmentation

Generally, the use of data augmentations were proven to be successful and improve the performance of the network. In the test data, some images had a low contrast which mean that the border between the Blood Pool and other structures was unclear. This would generally lead to quite bad segmentations, both manually and automatic.

The fact that the data augmentation included changes in contrast meant that the network had learnt to segment well on images with low contrast, resulting in good segmentations of these test images. The training data set did not include many images with low contrast, so this type of augmentation was important for the resulting network performance.

The distribution of the contrast augmentation parameter was changed with hopes that it would improve the performance. It was decreased, since it was thought that having too much contrast differences would decrease the performance. This did however not improve performance, and the distribution was later changed back to its original and larger interval values.

Changing data augmentations parameters sometimes led to better performance and sometimes to worse, and the reason behind this is not always according to theory. Flipping the images up/down was thought to not be necessary since upside-down images are not expected in practice. Because of this, the probability of flipping image patches like this was set to zero, but that was proven to worsen the performance. The reason for this is not clear, but changing a parameter back and forth and doing performance evaluation between changes allows for detection of these types of settings that are not always according to intuition.

For this reason, some other augmentations parameters where changed without much reasoning behind it, just to see if a good parameter combination could be found randomly. This was the case when changing the translation parameter, which was changed from 75% to 50%. It was also the case with the distribution of the noise, it was thought that increasing the distribution for both types of noise would increase the performance, since it would mean that the network would be trained on noisy and therefore "difficult" images. Increasing the distribution of streak noise was proven successful, but increasing the distribution of white noise was not, and no good explanation for this exists.

The distribution of scaling the image patches in the x- and y-direction was changed. The increase in distribution was motivated by the fact that giving the network image patches that were scaled with a bigger factor, would lead to better performance. This is because if the network is trained on more scaled images, it would be more robust to images data in different scales when it is used clinically. This change was proven to increase the network performance, which was according to theory.

7.5.3 Multi-task Segmentation

Including the class Bright was thought to improve the segmentation of the Blood Pool class. According to the evaluation metrics, the segmentations were approximately the same, but when visually evaluating the Blood Pool segmentations, it was clear that including Bright had worsen the performance. Without the Bright class, the Blood Pool was segmented well, with empty spaces for catheters, shunts and electrodes. With the class Bright, the Blood Pool did not always leave empty spaces for these structures and would instead classify them as Blood Pool, which was incorrect. The reason for this might be that during training, it was easier for the network to learn to classify the

bright structures as Background than Bright, since Background is kind of a "waste" class where everything left over is classified, and the class Bright is a class that has to compete for the pixels against other classes. In the end of the project, the class Bright was created when segmenting the training data, just so that these objects could be separated from the Blood Pool, and then deleted. But was not included in the actual network training data.

Including the class Bone did however improve the segmentations of Blood Pool. When including Bone in the training, the network learnt to identify features that were exclusive to, or at least predominantly occurring in Bone. By including this, the network learnt which structures were Bone and which were Blood Pool better, than when only Blood Pool was included.

The reasoning behind multi-task segmentation is that the learning process is improved if multiple classes is included if there is correlation between the classes. The fact that the class Bright did not improve segmentation, but the class Bone did, could be because there is no correlation between Blood Pool and Bright, but there is between Blood Pool and Bone. This could be because both Blood Pool and Bone include biological tissue, but Bright only includes artificial material, but most importantly that the Bright class was very uncommon in the data sets.

Bibliography

[1] Mended Hearts Organization, "Chd facts and statistics," [Online]. Available: https://mendedhearts.org/story/chd-facts-and-statistics/ (visited on 30/03/2022).

[2] H. Eliasson. "Barn med medfördda hjärtfel." (2021), [Online]. Available: https://www.hjart-lungfonden.se/sjukdomar/hjartsjukdomar/barns-hjartfel/ (visited on 31/03/2022).

[3] Skåne University Hospital. "Barnhjärtcentrum lund." (2018), [Online]. Available: https://vard.skane.se/en/skane-university-hospital/about-us/centres-of-excellence/pediatric-heart-surgery/ (visited on 31/03/2022).

[4] Team 3DSkåne, "About us," [Online]. Available: http://3dskane.se/about-us/ (visited on 10/05/2022).

[5] J. Dernelius, "3d-kopior av organ ger säkrare operationer," 2020. [Online]. Available: https://www.svt.se/nyheter/lokalt/skane/3d-organ-ar-framtidens-teknik (visited on 30/03/2022).

[6] 3DP, "Sjukhuset som printar egna implantat," 2021. [Online]. Available: https://3dp.se/2021/12/20/sjukhuset-som-printar-egna-implantat/ (visited on 10/05/2022).

[7] Medviso, "Segment 3dprint," [Online]. Available: https://medviso.com/segment-3dprint/ (visited on 12/05/2022).

[8] A. Bennström and F. Winzell, "Automated 3d bone segmentation using deep learning in scoliosis," M.S. thesis, Lund University, 2021.

[9] J. Zou, Y. Han and S.-S. So, "Overview of artificial neural networks," in *Artifical Neural Networks*, D. J. Livingstone, Ed. Totowa, NJ: Humana Press, 2008, pp. 14–22.

[10] TIBICO, "What is a neural network?," [Online]. Available: https://www.tibco.com/reference-center/what-is-a-neural-network (visited on 16/05/2022).

[11] H. Ya, L. T.Yang, Q. Zhang, D. Armstrong and M. J. Deen, "Convolutional neural networks for medical image analysis: State-of-the-art, comparisons, improvement and perspectives," *Neurocomputing*, vol. 444, pp. 92–110, June 2021.

[12] C. Camacho, "Convolutional neural networks," [Online]. Available: https://cezannec.github.io/Convolutional_Neural_Networks/ (visited on 16/05/2022).

[13] The MathWorks Inc. "What is a convolutional neural network?" (2022), [Online]. Available: https://se.mathworks.com/discovery/convolutional-neural-network-matlab.html (visited on 03/03/2022).

[14] IBM Cloud Education, "Convolutional neural networks," 2020. [Online]. Available: https://www.ibm.com/cloud/learn/convolutional-neural-networks (visited on 30/03/2022).

[15] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *CoRR*, vol. 2, 2015. arXiv: 1511.08458. [Online]. Available: http://arxiv.org/abs/1511.08458.

[16] Computer Science Wiki, "Maxpoolsample2," [Online]. Available: https://computersciencewiki.org/index.php/File:MaxpoolSample2.png (visited on 17/05/2022).

[17] O. Ronneberger, P. Fischer and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, 2015. arXiv: 1505.04597. [Online]. Available: http://arxiv.org/abs/1505.04597.

[18] N. Ibtehaz and M. S. Rahman, "Multiresunet : Rethinking the u-net architecture for multimodal biomedical image segmentation," in *Neural Networks*, T. Toyoizumi and D. Wang, Eds. Wako, Japan and Ohio,SA: Elsevier, 2020, pp. 74–87.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. 9, 2017. arXiv: 1412.6980. [Online]. Available: https://arxiv.org/abs/1412.6980.

[20] The MathWorks Inc. "Pixelclassificationlayer." (2022), [Online]. Available: https://se.mathworks.com/help/vision/ref/nnet.cnn.layer.pixelclassificationlayer.html (visited on 19/05/2022).

[21] ——, "Unetlayers." (2022), [Online]. Available: https://se.mathworks.com/help/vision/ref/unetlayers.html (visited on 19/05/2022).

[22] National Institute of Biomedical imaging and Bioengineering, "Computed tomography (ct)," [Online]. Available: https://www.nibib.nih.gov/science-education/science-topics/computed-tomography-ct (visited on 12/05/2022).

[23] E. Tiu, "Metrics to evaluate your semantic segmentation model," [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2 (visited on 31/03/2022).

[24] L. Baskaran, S. J. Al'Aref and G. Maliakal, [Online]. Available: https://doi.org/10.1371/journal.pone.0232573.g002 (visited on 17/05/2022).

[25] The MathWorks Inc. "Trainingoptions." (2022), [Online]. Available: https://se.mathworks.com/help/deeplearning/ref/trainingoptions.html#bu80qkw-3_head (visited on 17/05/2022).

Master's Theses in Mathematical Sciences 2022:E25
ISSN 1404-6342

LUTFMA-3474-2022

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/