Automatic Left Ventricle Segmentation of Long-Axis Cardiac MRI-Images

Ruben Bergengrip

Master's thesis
2021:E55

LUND UNIVERSITY

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

Automatic Left Ventricle Segmentation of Long-Axis Cardiac MRI-Images

Ruben Bergengrip

Supervisor: Einar Heiberg
Examiner: Karl Åström

Master's Thesis in Mathematical Sciences
Faculty of Engineering
FMAM05
2021

LUND UNIVERSITY

ABSTRACT

Over 30 % of all deaths worldwide are related to cardiovascular diseases [1] and they are the leading cause of death. Therefore, research into treatment, diagnosis and prognostication of these diseases are of critical importance.

One way to examine the cardiovascular system is through the use of Magnetic Resonance Imaging (MRI).

The aim of this thesis is to create a model using the Deep Learning architecture known as the U-net [2] to automatically segment the myocardium of the left ventricle of long axis cardiac MRI images. Using this model, the segmentations will be used to calculate the global longitudinal strain (GLS) of the left ventricle.

Several models were created and tested. In total, 601 subjects of varied age and sex were used to create and evaluate the models. The data was collected at several different sites using different vendors and settings.

The models were evaluated, and a final model was trained using the optimal settings and parameters. Using the final model, the strain was calculated for all images in the test set and the calculated strain was compared to the strain calculated using the manual delineations.

In conclusion, the final model was able to correctly segment the myocardium of the left ventricle in all views and time steps if given cropped images. The cropping can be done automatically or manually. The segmentations were able to be used for calculations of the global longitudinal strain with a coefficient of variation of about 4.6 % when compared to the manual delineations.

ACKNOWLEDGEMENTS

First and foremost, I wish to show my gratitude to my supervisor Einar Heiberg for his support and engagement in the project. In addition, I wish to thank the entire cardiac MR-group. I would also like to give an extra thanks to Amanda Nilsson, Andreas Bennström, Filip Winzell, Mathilda Larsson, and Julius Åkesson who have collaborated with me to improve my project and report. Finally, I would like to show gratitude to Anthony Lindholm and Jonathan Berg for manually delineating the data that was the basis for the thesis.

ABBREVIATIONS

Deep Learning

Adam – Adaptive moment estimation

ANN – Artificial neural network

CNN – Convolutional neural network

GPU – Graphics processing unit

ReLU – Rectified linear unit

Medical Imaging and Anatomy

AVPD – AV-plane displacement

CMR – Cardiovascular magnetic resonance imaging

CO – Cardiac output

CRT – Cardiac resynchronization therapy

CVD – Cardiovascular disease

EDT – End-diastole time

GLS – Global longitudinal strain

ICD – Implantable cardioverter defibrillator

LA – Left atrium

LV – Left ventricle

MRI – Magnetic resonance imaging

PAH – Pulmonary artery hypertension

RA – Right atrium

RV – Right ventricle

Statistics

CV – Coefficient of variation

FN – False negative

FP – False positive

IoU – Intersection over union

TN – True negative

TP – True positive

CONTENTS

INTRODUCTION

Cardiovascular diseases (CVDs) are the leading causes of death in the world and accounts for over 30% of all deaths worldwide [1]. In comparison, all forms of cancer account for about 17 % of deaths worldwide [3]. Risk factors for CVDs include hypertension, poor diet, a sedentary lifestyle as well as tobacco and alcohol habits. Since these diseases are so common, care and research are very important to reduce the number of deaths related to CVDs and improve the quality of life of those that suffer from CVDs.

Magnetic Resonance Imaging (MRI) is a medical imaging technique first developed in the 1970s [4]. It is primarily used to visualize anatomy, such as of the heart, but it can also be used to measure physiology and function, such as examining the pumping of the heart. Cardiac MRI, also known as Cardiovascular Magnetic Resonance imaging (CMR), is the use of MRI on the cardiovascular system (καρδία = heart) to assess function and anatomy. CMR-images are usually taken in one of two views, the long-axis view that bisects the heart along the long axis, and the short-axis view which bisects the heart along the short axis.

During this Master’s Thesis project, an automatic algorithm will be created to automatically segment, i.e. classify the pixels of, the heart muscle (myocardium) of the left ventricle of long-axis cardiac MRI images from 2-chamber, 3-chamber, as well as 4-chamber views (see Figure 1). This will be done using deep learning, specifically with the use of a U-net [2], and manually delineated data from the cardiac MR-group at Skåne University Hospital, Lund. If successful, applications for the usage of the segmentations will be explored as well.

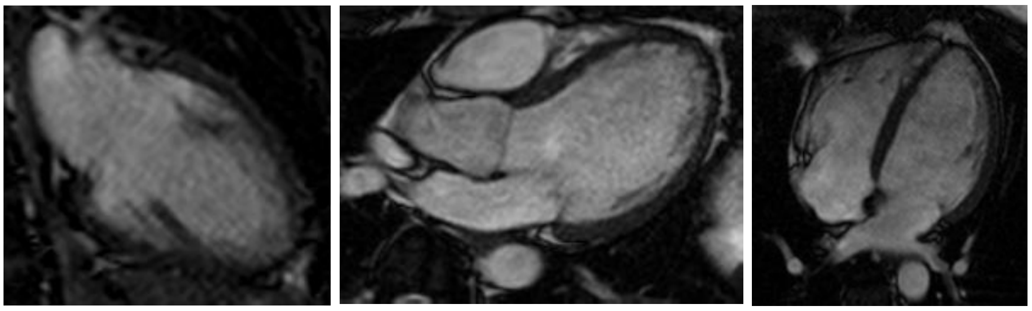

Figure 1: Long-axis CMR-images. From left to right: 2-chamber, 3-chamber, and 4-chamber views of the same heart. The left ventricle myocardium is the dark U-shape surrounding the heart (see Figure 2 for a manual delineation).

There are several reasons why segmentations of the left ventricle could be useful clinically. Segmentations of the left ventricular myocardium is needed in order to calculate the left ventricular global longitudinal strain which is used as an indicator of left ventricular systolic dysfunction [5]. Systolic function is how well the heart can contract and pump out blood. Using the same methodology, strain-rate can also be computed which is rate of deformation. This gives additional information on diastolic function of the heart. Diastolic function is how well the heart can fill with blood before the blood is ejected. In addition, the AV-plane displacement, i.e., how the plane that separates the ventricles and atria moves during the cardiac cycle, can be calculated which gives an indication regarding both the systolic and diastolic performance of the heart [6]. Both myocardial strain and AV-plane displacement have been shown to be important for prognosis of future cardiac disease [5] [6]. Finally, the ventricular volume can be estimated from the three long axis images. Ventricular volume and ejection fraction are important diagnostic markers for cardiac disease [7] [8].

Currently at the Skåne University Hospital, Lund, long-axis cardiac MRI images are not delineated clinically due to the time it takes for the delineations. However, the long-axis images are routinely collected and are therefore available for most patients. Manual delineations are currently only used in research studies. The manual delineations in this project are done using the software Segment by Medviso [9] (see Figure 2 for an example). This is performed three times, once for each of the three long-axis views. According to a previous study, the average time to manually delineate all three views is about 270 s with about 179 mouse clicks being needed [10]. If this would be done for all 30 time frames, the time would be over 2 hours assuming the user can keep the same performance during the entire time. Currently, a method known as feature tracking can be used to follow a manually delineated heart muscle through time [10] which makes it possible for the clinicians to only segment the end-diastole timeframe (EDT) instead of all, usually 30, timeframes for each view.

Feature tracking is useful for calculating the global longitudinal and radial strains; however, the performance of the feature tracking might not suffice for other potential applications such as AV-plane displacement or left ventricle blood pool estimation.

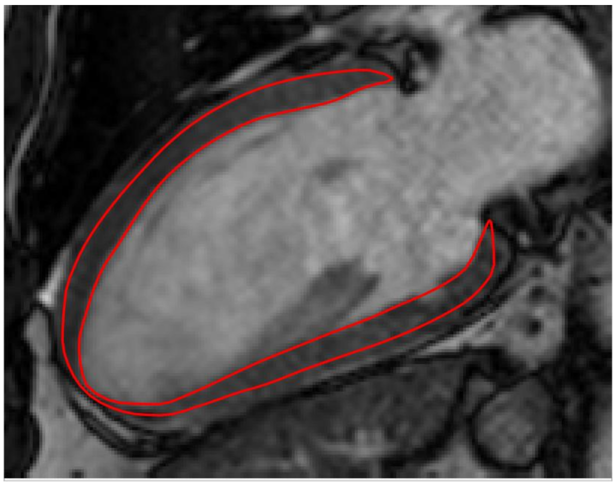

Figure 2: A 2-chamber CMR-image with a manually segmented left ventricle.

If an automatic method for delineation of the left ventricle existed, time and resources could be saved when calculating strains, which could lead to the clinical use of the global longitudinal heart since it has known prognostic uses. In addition, other measurements that require accurate delineations over all timeframes, such as calculating the AV-plane displacement as well as measuring the blood pool over time, would be far more feasible to perform regularly. If these measurements were more readily available, better diagnostics would be possible and more research could be performed on the importance of strain and strain rate and how it can be used prognostically in different diseases.

A potential challenge with the machine learning task is the diversity of the data. The data comprises a wide variety of patients of all ages and both sexes. In addition, the images have been collected over a significant period of time which leads to several different scanners and protocols having been used during imaging. As the images are taken from three distinct views, it is possible that three networks will have to be trained, one for each view, if the segmentation tasks for the different views differ too much. Finally, only the end-diastolic timeframe was delineated and usable for training and testing which could lead to poor performance on all other timeframes.

This Master’s Thesis has been made possible with the help of the Lund Cardiac MR group as well as Medviso AB who collaborate closely. The software Segment by Medviso was used for the delineations. The Cardiac MR group is a part of the Department of Clinical Physiology, Department of Clinical Sciences Lund, Lund University, Skåne University Hospital, Lund, Sweden, and they focus their research on cardiovascular pathophysiology and pumping.

AIM

The aim of the thesis is to segment the left ventricle of the heart from three different long-axis planes (two-chamber, three-chamber, and four-chamber) of cardiac MRI-images. Automatic segmentation will be performed using deep learning with a U-net architecture [2]. The result will be used primarily to estimate the global longitudinal strain of the muscle but also to estimate the AV-plane displacement. The resulting segmentation will be compared to manually delineated ventricles.

The model will be limited to only train and test on images from the end-diastole timestep. In addition, only the U-net architecture [2] with Adam [11] as its optimizer will be considered.

THEORY

Cardiology

The human heart is a complex muscle that is used to pump blood to the lungs and the rest of the body. Following the notation in Figure 3, the blood enters the heart in the right atrium (RA) from the caval veins, it is then pumped through the tricuspid valve (not visible) into the right ventricle (RV), and then through the pulmonary artery to the lungs where the blood is oxygenized. The blood is returned to the heart through the pulmonary veins and enter the left atrium (LA). The blood then passes through the mitral valve (not visible) into the left ventricle (LV) which is used to pump the blood to the rest of the body through the aorta. The atrioventricular plane is the plane that bisects the heart through the four valves, i.e., between the atria and ventricles. [12, pp. 74-76, 79]

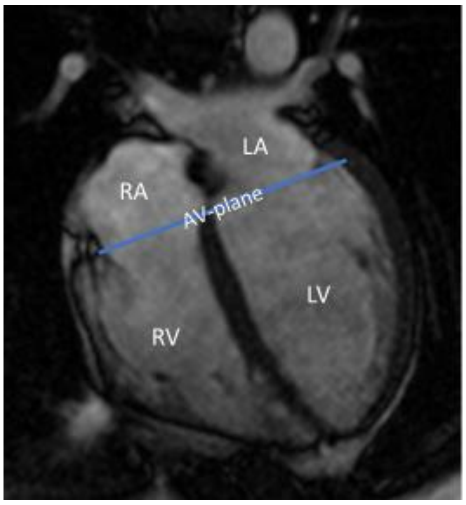

Figure 3: The anatomy of the human heart. The image is a 4-chamber CMR-image.

The heart cycle is divided into two phases, the diastole, and the systole [12, p. 110]. During diastole, the heart relaxes, while during systole, the heart is contracting and pumping the blood. The end-diastolic time is therefore the point in time just before the next contraction starts and it is when the ventricle is at its largest.

Heart failure is a condition that can arise from a number of different myocardial diseases such as infarctions, cardiomyopathies (i.e., diseases affecting the heart muscle) and hypertension (i.e., high blood pressure) [12, pp. 367-368]. The most common symptoms of heart failure are general fatigue as well as shortness of breath whilst exercising [12, p. 368]. The vague symptoms in combination with the increasing prevalence with age can make it difficult to discover and diagnose at first [12, p. 368].

According to a study by Willenheimer, Cline, Erhardt and Israelsson (1997) [6], the displacement of the atrioventricular plane during heart beats can be used as a predictor for mortality in heart failure. They concluded that the mortality was highly significantly correlated to the displacement, where a lower displacement was related to a higher mortality. The AV-plane displacement (AVPD) is used regularly in echocardiographic examinations [13] but usually not in cardiac MRI examinations [14].

Another measurement that can be taken using cardiac MRI-images is the global longitudinal strain of the heart. As shown by Park et al. 2018 [5] a decreased global longitudinal strain led to an increased risk of mortality for heart failure patients and should be used as a standard measurement for these patients. Strain is defined as the change in length divided by the original length. In the case of the heart, the longitudinal strain is defined as the strain along the entire length of the left ventricle. In the software Segment [9] the strain is calculated by tracking the distance between several points placed within the myocardium of the left ventricle and then tracked over the cardiac cycle (see Figure 4). Tracking is performed by non-rigid registration of subsequent frames and computing a deformation field.

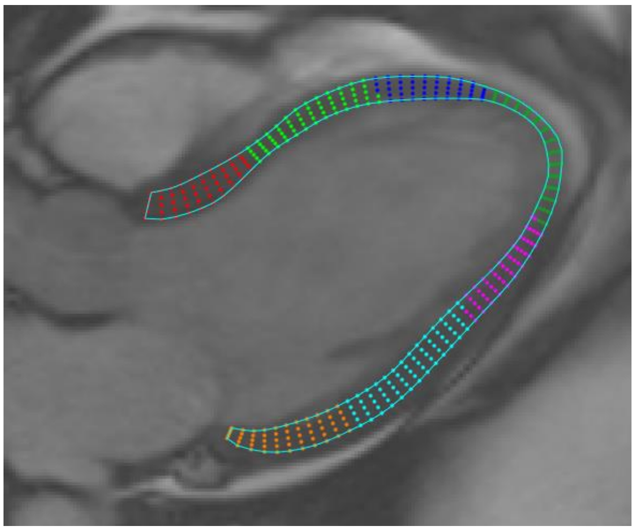
Figure 4: The points in the delineation that are tracked using Segment. The colours indicate different parts of the left ventricle for more regional calculations. The regions are the same as used in the 17-segment model [15].

By delineating the left ventricle, the blood volume of the left ventricle can be estimated [16]. This is often done using short-axis images but could theoretically be done using long-axis images as well. This can be advantageous for certain patients that have difficulties holding their breath for a longer time. If the patient has arrhythmias, this could interfere with short-axis imaging due to the number of slices needed for calculating the volume of the left ventricle. For these patients, the long-axis images could be useful for calculating the volume. In addition, it could be useful if only long-axis images have been taken of the patient.

Magnetic Resonance Imaging

Cardiac MRI is a technique where magnetic resonance imaging is used to image the heart. This can be used to non-invasively image the entire heart as well as assessing its function [12, pp. 69-70, 109]. One of the advantages of cardiac MRI is that it uses electro-magnetism instead of ionizing radiation which is used in X-rays, PET and SPECT, and is therefore not carcinogenic. There are only minor known health risks from the magnetic fields or the RF-pulses that are used during an MR examination [12, p. 55]. When someone is exposed to the RF-pulses, some of the energy will be transferred to the person as heat; this is very minor and not felt by the patient [12, p. 55]. Due to the risk of heating the tissue, there are guidelines regarding how much delivered energy and subsequent heating is acceptable for different body and tissue types [12, p. 55]. The more serious potential risks of the MRI are instead related to the magnetic object in the examination room that could be caught in the magnetic field of the machine [12, p. 54]. Examples of potentially dangerous objects are infusion stands, oxygen tanks, hospital beds and chairs. Another potential risk is the interaction between the magnetic field and medical devices, in particular implants, such as pacemakers and prostheses [12, pp. 58-60].

In MRI, the patient is placed inside a magnetic field created by an electromagnet. The hydrogen nuclei inside the body, primarily within water molecules, have a property called spin. By applying the magnetic field to the body, the hydrogen nuclei will align and precess (i.e., rotate) along the magnetic field due to this spin. The hydrogen nuclei can be pushed out of alignment using radiofrequency (RF) pulses similar to radio waves. By measuring the signal that is emitted from the nuclei when they return to their original state, an image can be created of the body depending on the hydrogen/water content of the different tissues. [12, pp. 1-3]

There are several settings that need to be chosen when acquiring an MRI image. The repetition time (TR) is the time between two consecutive RF-pulses [12, p. 13]. The trigger time is the time between the sent RF-pulse and when the data acquisition is begun [12, p. 17]. The echo time (TE) is the time between the RF-pulse and the peak of the echo response from the hydrogen nuclei [12, p. 13]. Finally, the flip angle is a measurement of how much the hydrogen nucleus will tip in its precession and is related to the amplitude and duration of the RF-pulse [12, p. 2].

In cardiac MRI, two axes are defined, along which images can be taken in different views [12, p. 96]. The short axis of the heart will bisect the heart as if slicing a loaf of bread, i.e., along the shorter width of the heart. On the other hand, the long axis would be if the loaf of bread were sliced along the "long" axis, i.e., the heart is sliced along the length instead. Examples of long-axis and short-axis CMR-images of the same heart can be seen in Figure 5. Short-axis images are used more commonly than long-axis images for calculating blood volume of the left ventricle. However, the long-axis images are routinely taken and used to visually assess the pumping of the heart. The global longitudinal strain, which has been shown to have prognostic value [5], could be calculated from the long-axis images as well as calculating the AV-plane displacement.

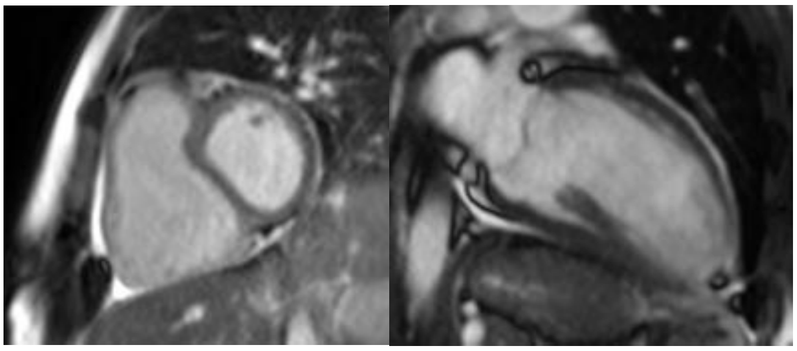
Figure 5: A short-axis (left) and long-axis (right, 2-chamber) image of the same heart.

When imaging the heart along the long axis, several planes are defined [12, p. 96]. The three imaging planes that are relevant for the thesis are the 2-chamber, 3-chamber, and 4-chamber planes previously shown in Figure 1. In the 2-chamber view, the left ventricle is bisected, and the left ventricle and atrium are shown. In the 3-chamber view, both ventricles as well as the left atrium can be seen. In the 4-chamber view, the left ventricle is bisected, showing both ventricles and atria.

Machine Learning

Artificial Neural Networks (ANNs) are a type of network that is used in machine learning. The ANNs are loosely based on the neural networks of the brain [17]. The ANN consists of several layers of nodes that are connected in specific ways. The input nodes are connected to nodes from a number of hidden layers and the nodes from the final layer are connected to the output layer that defines the output of the network [17] (see Figure 6). For each connection between nodes, there is a (usually trainable) weight attached [17]. The weights are updated according to a certain optimization function, such as stochastic gradient descent or Adam, based on the loss function [11]. If the network has many hidden layers, it is classified as a "deep" network. These are the networks used in Deep Learning [18, pp. 164-165].

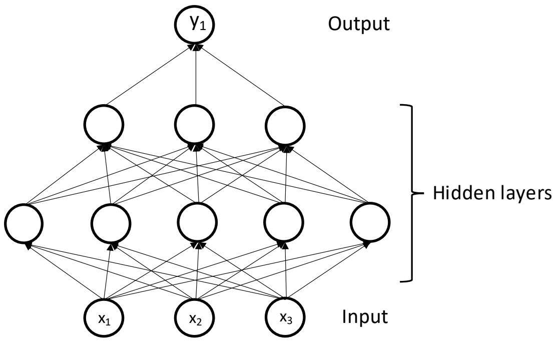

Figure 6: An ANN with an input layer, two hidden layers and an output layer. Used with permission from Andreas Bennström and Filip Winzell.

ANNs have proved to be very useful for a wide range of applications [19, pp. 1, 3]. In image analysis, ANNs are commonly used to classify images. This could be the classification of cell types, flowers, animals, or pretty much anything. A more advanced type of classification not only classifies an image, but it classifies and localizes objects within a single image. If every single pixel of an image is classified into a discrete set of predefined classes, it is called semantic segmentation [20].

Convolutional Neural Networks

One of the most common types of ANNs is the Convolutional Neural Network [18, p. 326]. The most important mathematical operation that is performed in the CNNs is convolution. The convolution is mathematically defined in equation (1) [18, p. 328]. In equation (1), *I* is the image and *K* is the kernel that is passed over the image. A practical example of how the convolution is calculated as well as the result from a convolution can be seen in Figure 7.

$$S(i,j) = \sum_m \sum_n I(m,n)K(i-m,j-n) = \sum_m \sum_n I(i-m,j-n)K(m,n) \quad \textbf{(1)}$$

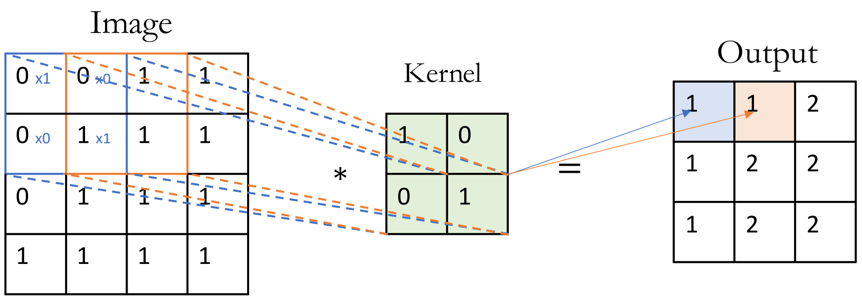

Figure 7: A convolution with a 3x3 kernel, 1 stride and no padding. Used and adapted with permission from Andreas Bennström and Filip Winzell.

In a CNN, each layer consists of a filter that transforms the image in a specific way. The most basic type of layer is the convolutional layer where a kernel, also known as a filter, commonly of size 3x3, is applied to the image [17]. The weights of the filter can be trained. Besides changing the size of the filter, the stride, i.e., how many “steps” the filter will take before being applied again to the image, can also be tweaked [17].

To avoid or minimize edge effects or ensure that the output has the same size as the input, different types of padding can be added to the original image. The simplest, and the one used during this thesis, is called zero-padding. Zero-padding, as the name suggests, pads the image with zeroes before the convolution is performed [17]. If the image is not padded, the kernel will probably not fit neatly over the image, leading to a reduced output image (such as in Figure 7), something that can be undesirable.

After the convolution, an activation function is applied to the result. Commonly, the ReLU, or rectified linear unit, is used. The ReLU activation function can be seen in equation (2) [21]. There are a couple of reasons why ReLU is used. First, since it is almost linear, it has several of the benefits of regular linear models that allows them to generalize well [18, p. 170]. In addition, since the ReLU isn’t quite linear, it avoids the problem of the entire network reducing down to a linear combination of all layers [18, p. 170].

$$f(x) = \max(0, x) \quad (2)$$

Another common type of layer in the CNN is the max-pooling layer which is used to reduce the image size as well as allowing the following kernels to learn broader features and provides a more translationally invariant model [18, pp. 335-336]. Using max-pooling, a kernel, usually a square of size 2x2 or 3x3, is applied to the image with a certain stride. For each stride, the largest value that the kernel encompasses is passed through to the resulting image while the other values are discarded (see Figure 8). When a larger stride is used, more of the image is discarded and the spatial dimensions are reduced even more.

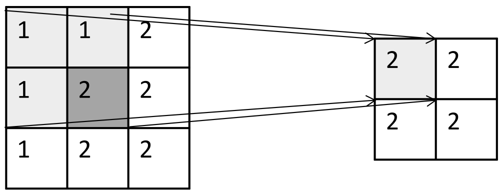

Figure 8: An example of max-pooling with a 2x2 kernel and stride 1. Used and adapted with permission from Andreas Bennström and Filip Winzell.

Up-convolution is a similar but opposite operation as the max-pooling. Instead of decreasing the image size, the up-convolution will increase it and interpolate the added values [22]. A visualization of this can be seen in Figure 9.

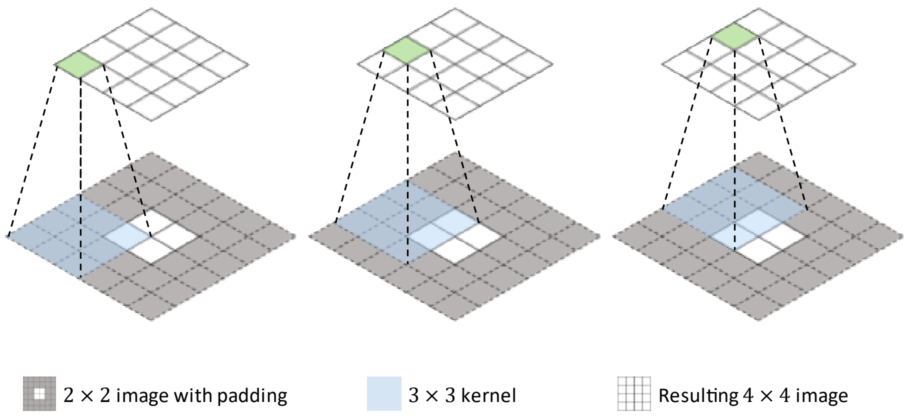

Figure 9: A visualization of up-convolution.

There are several advantages with using a CNN over other types of neural networks. CNNs usually have very sparse connectivity; this means that, due to the use of a kernel, the pixels from the first layer are only connected to a limited number of kernels in the resulting output image [18, p. 330]. This reduces the number of weights that are needed tremendously. In addition, the weights are shared between the different pixels since the kernels do not change values for its weights while moving [18, pp. 331-333]. This will also reduce the number of needed weights. In total, if two layers of 128x128 images would be fully connected with no shared weights, over 15,000 weights are needed. However, if a CNN is used with a 3x3 kernel, only about 10 weights are needed instead. This will improve both the efficiency as well as reducing the computational memory and requirements needed to run and train the model [18, p. 330].

U-net Architecture

The U-net architecture described by Ronneberger, Fischer and Brox (2015) [2] is a deep learning architecture that has been shown to produce excellent results for semantic segmentation of biomedical images [2]. A schematic of the architecture can be seen in Figure 10. There are four main operations in the U-net generally going in different directions in the figure. The operations going downwards, the contractive path, are max-poolings with a 2x2 filter with stride 2, i.e., the image size is reduced. In an opposite manner, the operations going upwards, the expansive path, are 2x2 up-convolutions that increase the size of the image. The two remaining operations are going across the U. The unpadded 3x3 convolutions with ReLU activation function are performed twice before the up-convolutions and the max-poolings respectively. At the same time as the max-pooling is performed, a copy of the current image before max-pooling is made. This copy is cropped to match the size on the opposite side of the U and is added as additional image channels on the opposite side; this is called a skip-connection. The final layer consists of a 1x1 convolution designed to map the resulting image to the number of classes that are to be identified. The depth of the U-net, i.e., the number of layers, can be tweaked by the user and the standard size described in the article [11] is 4.

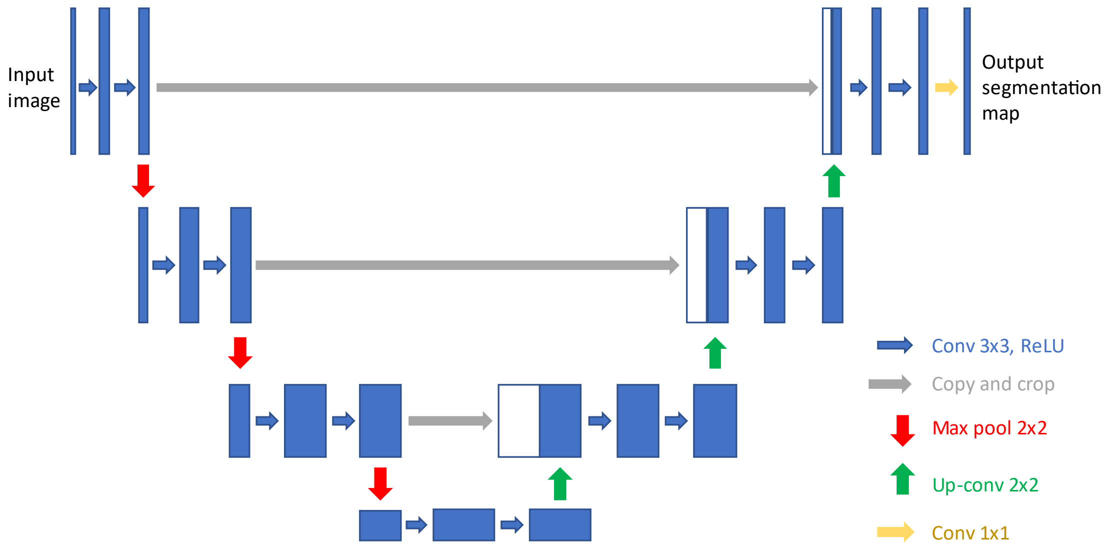

Figure 10: A schematic view of the U-net architecture. The contracting pathway can be seen on the left side and the expansive can be seen on the right. Used with permission from Andreas Bennström and Filip Winzell.

Loss functions

To be able to update the weights in the network, a loss function is needed. The loss function will calculate how well the current weights perform on the data, that is how well the output of the network corresponded to an ideal output given the input. A higher loss is worse, and an optimizer uses the value of the loss function to update the weights in the network to minimize the loss. The weights are updated using the gradient of the loss function with respect to the weights. Using the chain rule of calculus to back-propagate through the layers of the network, the gradient with respect to all weights can efficiently be calculated [18, pp. 200-204].

The loss can be calculated in several different ways. The most common loss function is the cross-entropy error. The cross-entropy is calculated as in Equation (3), where y is the target and p is the prediction [23]. A weight can also be added to each term in equation (3) to create a weighted version of binary cross-entropy [23].

$$L_{CE}(y,p) = -(y\log(p) + (1-y)\log(1-p)) \tag{3}$$

Another loss function is the Dice-loss which uses the Dice-score (see Evaluation metrics below for more details) as a basis for the loss [23]. This can be useful in semantic segmentation since the Dice-score and similar metrics are often used to evaluate the results from the segmentation. The Dice loss can be seen in equation (4).

$$L_{DL}(y,p) = 1 - \frac{2yp+1}{y+p+1} \tag{4}$$

A combination of cross-entropy and Dice loss has also been used where the loss was the weighted average of the two losses as seen in equation (5) [24]. α is a value between 0 and 1.

$$L_{Combo} = \alpha L_{DL} + (1-\alpha)L_{CE} \tag{5}$$

Optimization

The optimizer of the neural network determines how the weights of the network will be updated based on the loss. One of the most popular optimization algorithms is called Adam, as described by Kingma and Lei Ba in 2015 [11]. There are three hyperparameters, i.e., parameters that control the learning, that can be tweaked when using Adam, though two of them are often set to their standard values. The learning rate determines how much the weighs will be updated and is usually set to a small positive value. The learning rate depends heavily on the data and has to be tweaked. The other two hyperparameters that are relevant for Adam are called β_1 and β_2. The β_1 and β_2 values are designed to control how much different derivatives are used when updating the weights and they are modelled after other optimizers such as Stochastic Gradient Descent and Root Mean Square Propagation. The default settings are β_1 = 0.9 and β_2 = 0.999 [11].

Besides the parameters related to the optimizer, there are several other hyperparameters that need to be chosen. The number of epochs describes how many times the entire training dataset will be sent through the network when training [25]. A higher number of epochs gives the network more chances to learn new things from the data with an increased risk of overtraining the model, a concept that will be discussed below in Generalization performance. The minibatch size determines the size of the subset of your data that will be used in one iteration, i.e., before the weights are updated [18, p. 275]. The minibatch size is often limited by the GPU memory since larger minibatch sizes require more memory [18, p. 275].

Generalization performance

The goal of any machine learning algorithm is to achieve high generalization performance. The generalization performance is the performance on data that the model has never seen before [18, p. 108]. The generalization performance is generally estimated by separating the training data from a set of testing data. The testing data should only be tested on once to avoid adding bias to the model. This leads to another problem since we need to test the model regularly when optimizing different parameters to ensure that we use the optimal values. This is solved by using a third dataset, the validation dataset, that is used during training and hyperparameter tuning while the testing set is saved for testing the final model(s) [18, p. 119].

A better method for estimating generalization performance, especially when using a smaller dataset, is to only split the data into a large training dataset and a smaller testing dataset. The larger training dataset will then be further split into *k* parts, known as folds. Following this and comparing with Figure 11 for a visualization, *k* different models will be trained using *k*-1 folds and validating on the final fold. The average validation performance can then be calculated to get a better estimate of the generalization performance than if only splitting the data once. When the optimal parameters have been found, a new model is trained using all *k* folds as training data and testing on the test dataset. This method is called *k*-fold cross-validation [18, p. 120].

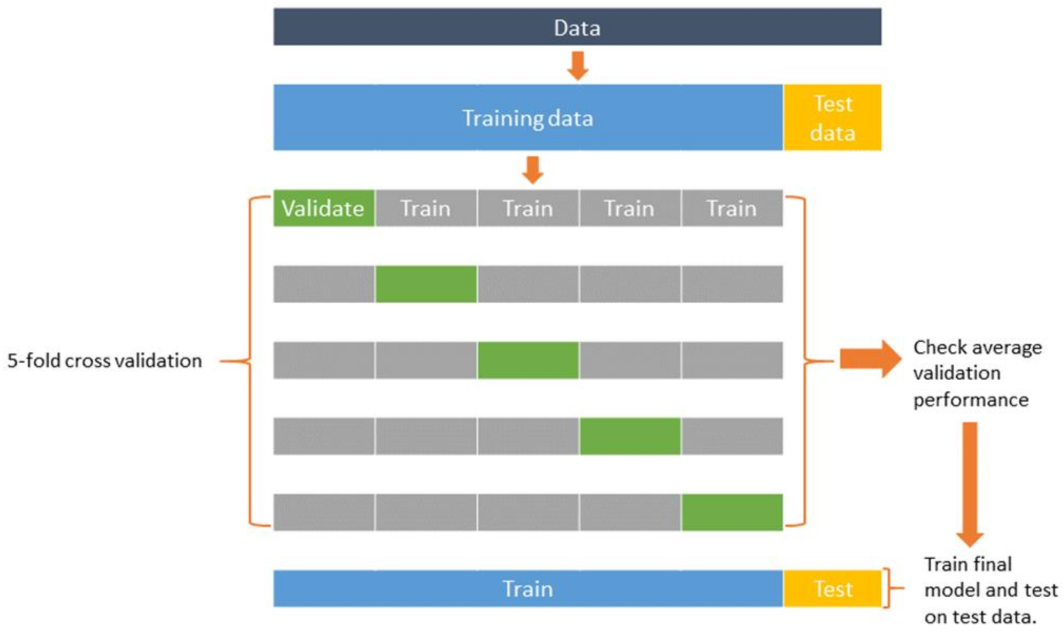

Figure 11: A visualization of k-fold cross validation with k=5.

When training the machine learning model, there are two primary goals to ensure good generalization performance. The first goal is to minimize the training error to ensure that the optimal solution has been found. If the training is stopped too early, or the model is not capable of fitting to the data well enough, the model will be underfitted (see Figure 12) [18, pp. 109-110]. The second goal is to minimize the difference between the training and validation errors. If the training error continues to decline while the validation error stays the same, or even worse, increases, it's a sign of overtraining (see Figure 12) [18, pp. 109-111]. Overfitting is what happens when the model starts to learn peculiarities from the training dataset in an effort to minimize the training error that won't generalize well to unseen data. In image analysis, an example could be that a model realizes that all images that include blue sofas also happen to include cats and therefore always classifies images with blue sofas as cats. This will reduce the training loss (since all images with blue sofas will be correctly labelled as cats) but will increase the validation and test loss since blue sofas do not actually have anything to do with cats. The likelihood of overtraining increases when the capacity, or complexity, of the model increases as well as when the amount of training, i.e., the number of epochs, is increased [18, p. 110].

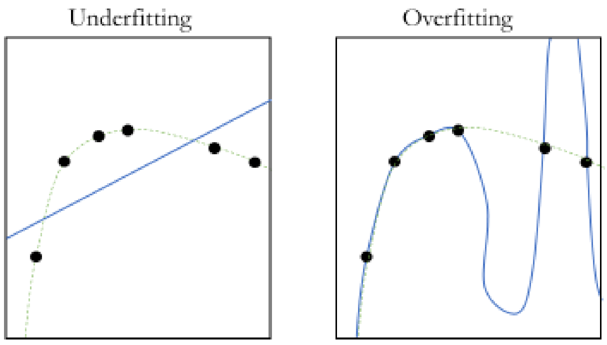

Figure 12: A visualization of underfitting and overfitting, respectively. The dashed line is the optimal fit for the datapoints. Used with permission from Andreas Bennström and Filip Winzell.

There are several techniques to decrease the losses without accidentally overtraining the model when, for example, the dataset is very small, something that is very common in biomedical image analysis. The U-net architecture has been shown to be very successful for segmentation tasks even when only a small dataset is available by extensive use of image augmentations [2]. Image augmentation is a technique used in deep learning to change the training dataset in a way to allow the network to learn more from each image without having to add more data. There are a couple of different types of image augmentation. Spatial augmentations include rotation, mirroring, shearing and translation [26]. The applicability of these augmentations is heavily dependent on the dataset, for example would mirroring not be useful for classifying letters or numbers due to the nature of their shapes. Another type of augmentation relates to the image quality [26] and includes, among others, sharpness, blurriness, and the level of noise. Finally, image appearance can by augmented by changing the brightness, contrast, or similar settings [26].

Another technique used to decrease the risk or impact of overtraining is the usage of different regularization functions. One of the more common is the L2-regularization. When using L2-regularization, a term, called the L2-regularization term, is added to the loss function. This term is the squared sum of all weights (denoted ω in equation (6)) in the network and will reduce the size of weights by giving them a larger penalty than smaller weights (see equation (6)) [18, pp. 226-227]. This is advantageous since larger weights are needed for the model to overtrain, something we want to avoid. The L2-regularization parameter can be tweaked by changing λ.

$$\Omega = \frac{\lambda}{2}\sum_i {\omega_i}^2 \tag{6}$$

Evaluation metrics

To determine the performance of a deep learning model, it is important to know which evaluation metrics can be used and what they mean. To understand the metrics, the terms shown in the confusion matrix in Figure 13 are important to understand. As seen in Figure 13, the true positives (TPs) and true negatives (TNs) are the correct predictions by the models. In an opposite manner, the false positives (FPs) and false negatives (FNs) are the incorrectly predicted pixels [27].

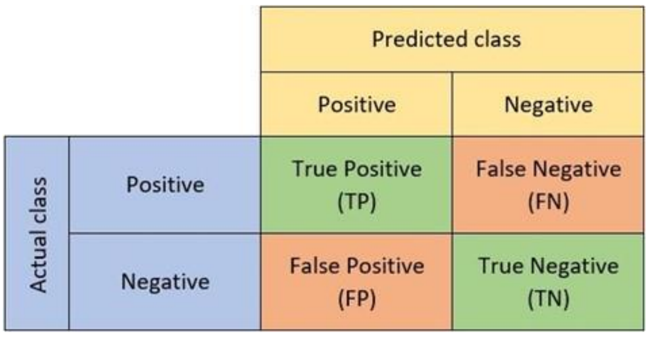

Figure 13: A 2x2 confusion matrix showing the definitions of a true positive, true negative, false positive, and false negative result. Used with permission from Andreas Bennström and Filip Winzell.

The most common evaluation metric is the accuracy which is given in equation (7). The equation is simply the number of correctly classified pixels divided by the total number of pixels in the image. This metric is significantly more useful for balanced datasets than unbalanced datasets. If 90 % of a certain dataset contains a single class, the accuracy will be 90 % if the algorithm decides to classify everything as the dominant class. Note that the accuracy for each class can be calculated independently, thus reducing the problem.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

A metric that is often used for determining the success of a segmentation, is the dice score, also known as the Sørensen–Dice coefficient. The equation for the dice score can be seen in equation (8), where y is the ground truth and p is the prediction. The Dice-score is less sensitive to class imbalances than the accuracy and is more useful for determining the performance of segmentations [28].

$$D(y,p) = \frac{2|y \cap p|}{|y| + |p|} \tag{8}$$

Finally, the intersection over union (IoU), also known as the Jaccard index, can be used for a similar purpose as the Dice-score [20]. The IoU score is calculated by dividing the number of overlapping pixels between the ground truth and the prediction (the intersection) with the number of total pixels in the prediction and ground truth (the union). Mathematically, this can be written as in equation (9) where y is the ground truth and p is the prediction [28]. An intuitive visualization over how the IoU is calculated can be seen in Figure 14.

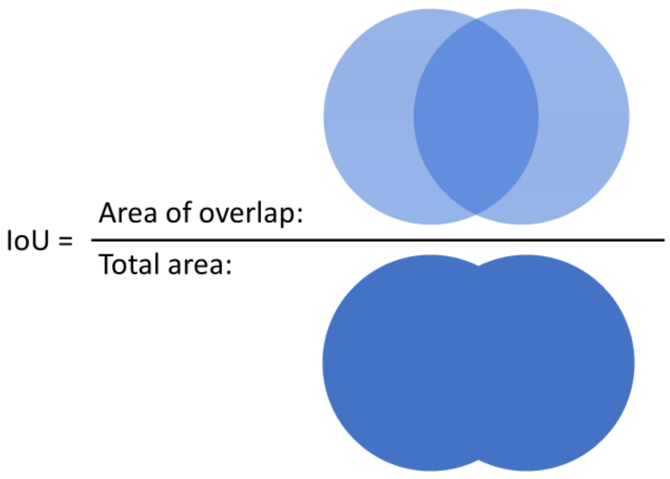

Figure 14: The definition of intersection over union (IoU) visualized. The intersection is the area of the overlap. The union is the total area.

$$J(y,p) = \frac{|y \cap p|}{|y \cup p|} \tag{9}$$

DATA

The data consisted of cardiac MRI images from 601 subjects that came from six different sources. The different sources are listed below with some additional information surrounding the type of patients that were part of each study.

- The AL dataset included patients from both Sweden and Greece and included patients with systemic sclerosis [29].
- The CHILL-MI dataset came from a study that examined the use of central venous catheter core cooling combined with cold saline [30]. The patients from this study all had myocardial infarctions a couple of days prior to the examination. In addition, some patients were scanned again after six months. The patients were scanned at 9 different sites from 4 countries.
- The PAH (Pulmonary Arterial Hypertension) dataset came from two different studies that included healthy controls, patients with pulmonary hypertension as well as patients with systemic sclerosis without PAH [31] [32].
- The CO (Cardiac Output) dataset [33] was more varied as it included healthy volunteers, athletes as well as patients with congestive heart failure.
- The CRT (Cardiac Resynchronization Therapy) dataset [34] contained patients with various cardiovascular diseases.
- The ICD (Implantable Cardioverter Defibrillator) [35] dataset contained patients with different kinds of cardiomyopathies.

The patients had a variety of pathologies since the data was collected for a wide variety of reasons. The data was pseudonymized before being given to the author. The images were acquired over a long period of time, the earliest image was taken in 2003 and the latest in 2017, using both Siemens (19 %) and Philips (81 %) scanners. The patients were of varied gender and age. 363 patients were male, 120 were female, and the remaining 118 were removed in the pseudonymization process. The mean age was 59.3 and the standard deviation was 13. The youngest patient with age data available was 5 years old and the oldest was 87. For the MRI settings, the echo time was typically around 1.47 ms, the repetition time was around 3 ms, and the flip angle was 60°. The image resolution was typically 0.75 mm.

Most of the datasets included 2-chamber, 3-chamber, and 4-chamber views but some lacked one of the views. In addition, the image stacks included images from one complete heart cycle, usually 30 images, but since only the end-diastole phase was segmented, this was the only image that was used for training from each view for each patient. In addition, all images had been manually cropped to only include the heart. The segmentations were performed with the software Segment from Medviso AB, Lund, Sweden [9] and were delineated by Anthony Lindholm (for the AL, CHILL-MI and PAH datasets) and Jonathan Berg (for the CO, CRT and ICD datasets).

METHOD

In the following section, the methods and materials used during the thesis will be discussed. The method will be divided into the pre-processing of the data, the model selection and settings, the different tests that the models were used for, as well as the evaluation methods that were used to determine the quality of the models.

The programming tasks were completed using the software MATLAB [36] by MathWorks and Segment by Medviso [9]. The networks were trained using two NVIDIA Titan RTX GPUs as well as two NVIDIA GeForce RTX 3090 GPUs that were made available by the cardiac MR-group.

Splitting the data

The data was split twice, once for the initial hyperparameter tuning and once for the final tests. For the hyperparameter tuning, the data was divided into 70 % training, 15 % validation and 15 % testing as visualized in Figure 15. The data was divided by randomly choosing patients for each of the seven sources independently, i.e., roughly the same number of 2-chamber, 3-chamber and 4-chamber images will be found in each of the sets. In addition, since the same 70/15/15-split was performed independently on the different sources, the potential bias from the different sources was minimized.

Training
Validation
Testing

Figure 15: The training, validation, testing split visualized.

For the final tests that will be presented in the results, another split was used. The 15 % test data was kept the same as for the hyperparameter tuning. However, the remaining 85 % was divided differently. Instead of dividing it into training and testing, it was divided into 5 equal folds in accordance with 5-fold cross-validation as was discussed earlier. See Figure 11 on page 15 for a refresher. The final tests were performed using 5-fold cross-validation and the average validation performance over the five folds was used for comparing the different networks. The final model was then trained using the best settings on all 5 folds and was then tested on the test set.

Artificial edge class

For most tests, the ground truth masks were pre-processed in a very specific way. Originally, the masks included two classes, the background, and the left ventricle. An artificial third class was introduced by modifying the existing masks. This artificial class was created by dilating the original masks using a square with a side of 10 pixels. The new pixels that were added due to the dilation were labelled as the artificial class while the pixels from the original mask were kept as the ground truth for the left ventricle. This new class will from here on be called the “artificial” class or the “edge” class. To the author’s best knowledge, the usage of an artificial edge class is a novel idea. However, the edge class can be compared to a simplified, truncated distance map similar to the work of Åkesson [37].

Since the artificial class is treated the same as the other classes by the network, it is possible to weigh it differently than the background and foreground to optimize the loss function. However, since this was not needed in this case, only the standard weighing of setting the weights inversely proportionally to the number of pixels was used. An example of a mask with the artificial class can be seen in Figure 16.

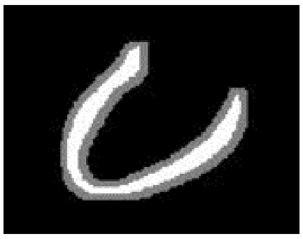

Figure 16: A mask from a 2-chamber CMR-image with the artificial class in grey.

Input normalization

Before the data was fed to the U-net, input normalization was performed on all data. Two types of input normalization were performed depending on the desired MATLAB version compatibility. For networks compatible with MATLAB 2019a, the data was only normalized by setting the mean to zero. However, for the MATLAB 2020a compatible network, z-scoring was used. This normalized the data to have zero mean as well as a standard deviation of 1.

Model selection

When selecting the model, some hyperparameters and related settings were already predetermined such as the architecture being a U-net and the optimizer being Adam. The remaining hyperparameters were tested in a way inspired by the grid search [18, pp. 427-430], i.e., different combinations of values were tested and the combination with the best results were chosen. The values seen in Table 1 represent the values that were kept consistent for most or all of the final tests. The learning rate drop period is the number of epochs before the learning rate is reduced. The learning rate drop factor determines how much the learning rate is dropped after each period.

Table 1 shows the values for the hyperparameters that were used.

Hyperparameter	**Value**
Initial learning rate	5e-4
Learning rate drop factor	0.1
Learning rate drop period	200
L2 regularization parameter	1e-6
Max epochs	700
Mini-batch size	32
U-net encoder depth	4

Several different loss functions were tried but it was decided that weighted cross entropy would be used. In addition, Dice loss, unweighted cross-entropy loss as well as combo loss were also tested. The weights for the weighted cross-entropy were set as the median of the frequencies for the classes over the frequency for each class. This turned out to be about 1.68 for the left ventricle, 1 for the artificial class, and 0.1129 for the background. This means that there was about 17 times more background pixels (including the artificial class) than foreground pixels.

Data augmentation

Several augmentations were performed on-the-fly as the networks were trained. All augmentations were performed on all samples and the values for the augmentations were taken from a uniform distribution. The augmentations that were used were in part inspired by the work of Isensee et al. [38].

The spatial augmentations that were performed were rotations, translations, reflections, scaling, and shearing. All spatial augmentations were performed along both the X-axis as well as the Y-axis. The values that were used can be seen in Table 2.

In addition to the spatial augmentations, three additional augmentations were performed. Gaussian blur was added using a Gaussian smoothing kernel with a standard deviation (σ) that was taken from a uniform distribution between 0 and 1.5. The filter size was set to $2 \cdot \mathrm{ceil}(2\sigma) + 1$. Similarly, sharpening was applied in the same way with the same magnitude. The brightness of the images were augmented as well by simply multiplying the pixel intensities with a value taken from the uniform distribution U(0.5,1.4). See Table 2 for the complete list of augmentations. Sharpening and blurring occur with a probability of 10 %, the remaining augmentations are always performed.

Table 2 shows the magnitudes of the augmentations that were used.

Augmentation	**Value**
Rotation	U(-30°,30°)
Translation in X and Y	U(-10,10)
Reflection in X and Y	True/False
Scaling	U(0.95,1)
Shearing in X	U(-30°,30°)
Shearing in Y	U(-30°,30°)
Gaussian blur	U(0,1.5)
Sharpen	U(0,1.5)
Brightness	U(0.5,1.4)

One final augmentation was tested in the preliminary tests. This augmentation took the manual segmentation from the end-diastole time-step and used a feature tracking algorithm to generate training data for all other time-steps. This led to a thirty-fold increase in the training size. This was only used for one model.

The augmentations that were used were chosen by both experimentally confirming that they improved the validation performance as well as by using logical reasoning surrounding what could differ between different datasets. Since all images were manually cropped, variation due to translation and scaling, i.e., where and how much the images were cropped, should exist. Due to the nature of how the images were taken and the human anatomy, some variation in rotation should be expected. Since the images were oriented in different ways it made sense to use reflection as an augmentation method as well. Shearing was applied to account for differences in how the heart was shaped and to account for deformation of the heart when pumping. Finally, Gaussian blur, sharpening, and brightness augmentations were used to simulate images being taken by different scanners or using different setting.

Strain calculations

The final models were then used to perform strain calculations. The strain calculations were performed in the software Segment using the same method as for the datasets that were manually delineated for the best comparison. The mean global longitudinal strain was calculated for each view separately using the delineations from the model and was compared to the strains calculated using the manual delineations.

A plugin for Segment was created to perform the strain calculations in an efficient manner. Using the plugin, the user could choose which (or all) view planes to segment as well as being able to choose if only the end diastole timeframe should be segmented or if all timesteps should be segmented. For the strain calculations, only the end diastole time frame was segmented and the rest of the timeframes were segmented using the pre-existing feature tracking method. In addition to using the network to produce the segmentation, smoothing was performed before the segmentation was used for the strain calculations. The smoothing was performed using Gaussian smoothing with a kernel of size 0.8. A closing operation was performed with a disc of size 10 to increase the likelihood of the segmentation being continuous. Since only a continuous border can be used for the strain calculations, the largest object from the resulting smoothing and closing was kept in the cases where more than one object remained. The contour of this object was extracted and exported to Segment.

Testing on uncropped data

Some additional tests were performed on non-delineated uncropped images. The images were sent to the model uncropped and were then manually inspected to assess the performance. In addition, two different cropping methods were tested as well. The first cropping method was manual, where the user chose how to crop the image. This feature was already implemented in Segment. In addition to this, an automatic cropping method was created, where a preliminary segmentation was performed, and the image was cropped around the preliminary segmentation before another segmentation was performed. The cropping is done using a rectangle positioned at the most extreme coordinates for both the x- and y-axis and adding a buffer zone of 10 % of the total size in x- and y-directions respectively. The cropping is skipped if the largest coordinates in either x or y is larger than 80 % of the total length of that axis.

AV-plane displacement

In addition to the strain calculations, a script was made to calculate the AV-plane displacements. The script found the AV-points as well as a reference point on the opposite side of the heart to be able to calculate the orthogonal distance from the point to the AV-plane (see Figure 3). The reference point was found by finding the maximum total distance between every point in the segmentation and the two AV-points. The reference point was fixed for all time frames. By calculating the distance between the AV-plane and the reference point at every time frame for all views, the displacement was found and plotted as a function of time.

Other timesteps

Other timesteps than the end-diastole time were segmented and manually inspected, but due to the lack of manually delineated data for other timesteps than the end-diastole time, no exact tests were performed on them.

Evaluation methods

The models were primarily evaluated by assessing the automatic segmentation in comparison with the manual segmentation. The delineations were compared using the IoU score (equation (9) and Figure 14) with a strong emphasize on the IoU for the left ventricle. In addition, paired t-tests were performed to check whether the improved results between the best model and the other models were significant.

Since no manual delineations of other timesteps than the end-diastole timestep existed, these methods could not be used for evaluating the performance on other timesteps. For this reason, only visual inspection was performed on the other timesteps.

Strain

The results for the strain calculations were assessed using Bland-Altman plots [39] as well as scatter plots. The scatter plots plotted the predicted strain against the ground truth with points closer to the identity line being better. The Bland-Altman plots plotted the difference between the prediction and the ground truth on the y-axis and the mean of the two methods on the x-axis. In this case, a horizontal line around $y = 0$, i.e., no difference between the prediction and the manual segmentation, is the optimum.

AV-plane displacement

Due to time constraints and lack of manually delineated data for AVPD, the created script for calculating the AVPD was not compared to anything and was only evaluated visually.

RESULTS

In this section, the results from the tests will be presented in the following order: first, the different models and hyperparameter values, then the strain results, and finally any other miscellaneous tests.

When training a model using five-fold cross-validation, it took about 15-20 hours to train and validate all five folds. Usually, two cross-validations were run in parallel on a single GPU.

An example of a training plot using Dice loss can be seen in Figure 17.

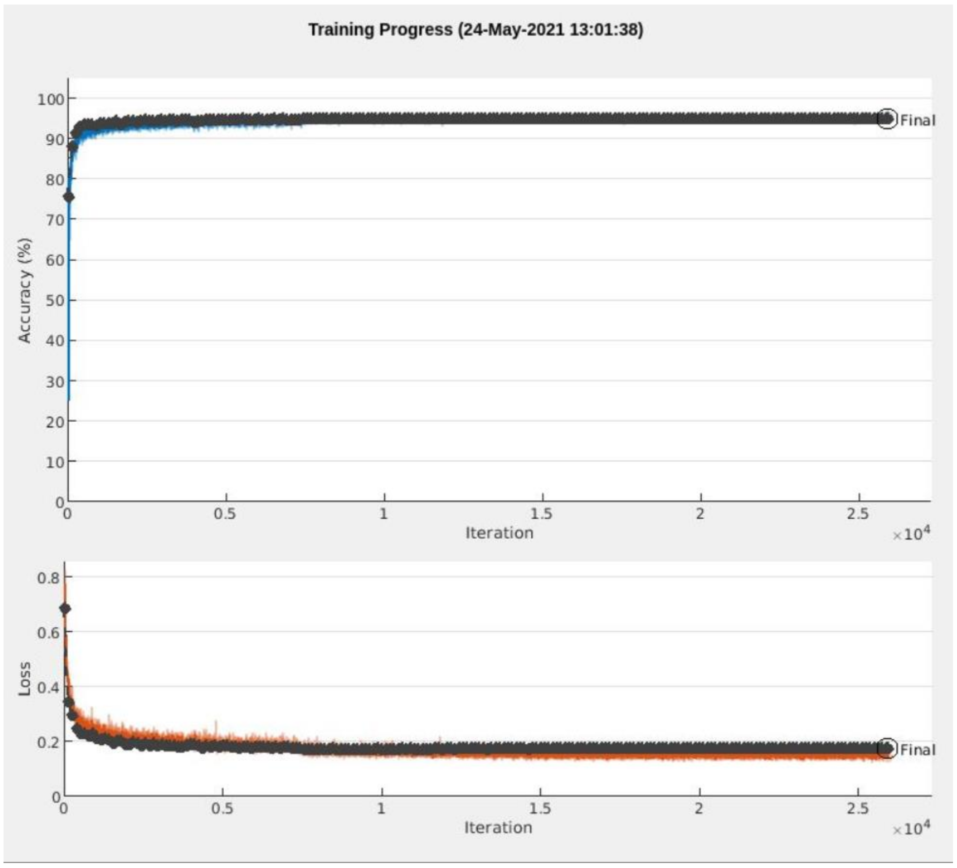

Figure 17: A training plot showing the accuracy (top) and loss (bottom) for using Dice loss. The black dots are the validation performance, and the lines are the training performance.

Model performance

Six different loss functions were tested and compared in Figure 18. The best loss function was determined to be the weighted cross-entropy since it had the highest mean IoU over all five folds ($\mu = 67.3$). However, most of the loss functions gave similar results. Nevertheless, the weighted cross-entropy is probably better than the Dice loss ($p = 0.052$) and definitely better than the unweighted cross-entropy ($p = 0.014$).

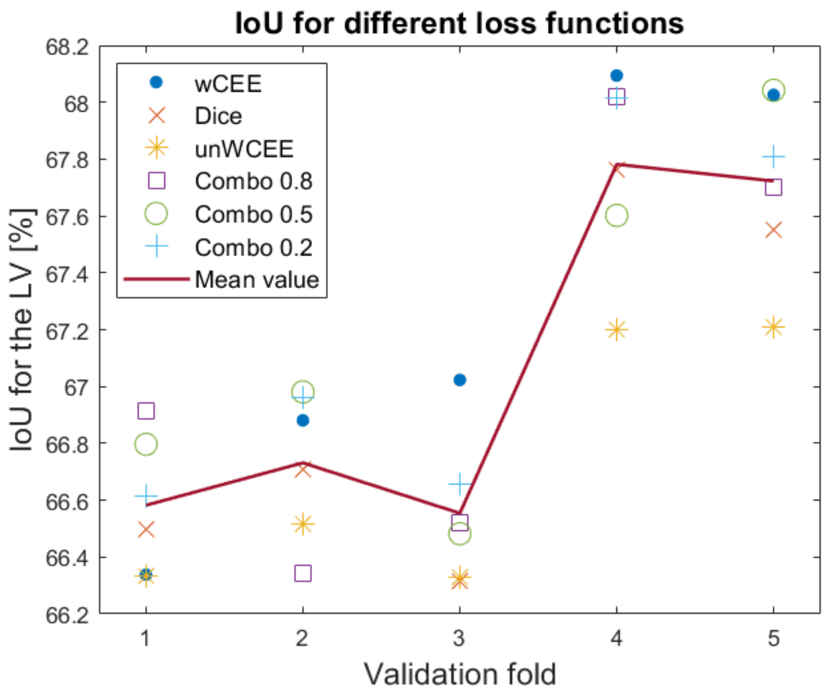

Figure 18: Comparing the results for different loss functions. Here, weighted cross-entropy error (wCEE), Dice loss, unweighted cross-entropy error (unWCEE) and combo loss with three different alpha-values (0.8, 0.5 and 0.2) were compared. See equation (5) regarding combo-loss and alpha-value.

Two different U-net network depths were tested, 4 and 5, and the results can be seen in Figure 19. Depth 4 was not statistically significantly better than depth 5 ($p = 0.30$) even though the mean value was slightly higher (67.3 vs 67.1).

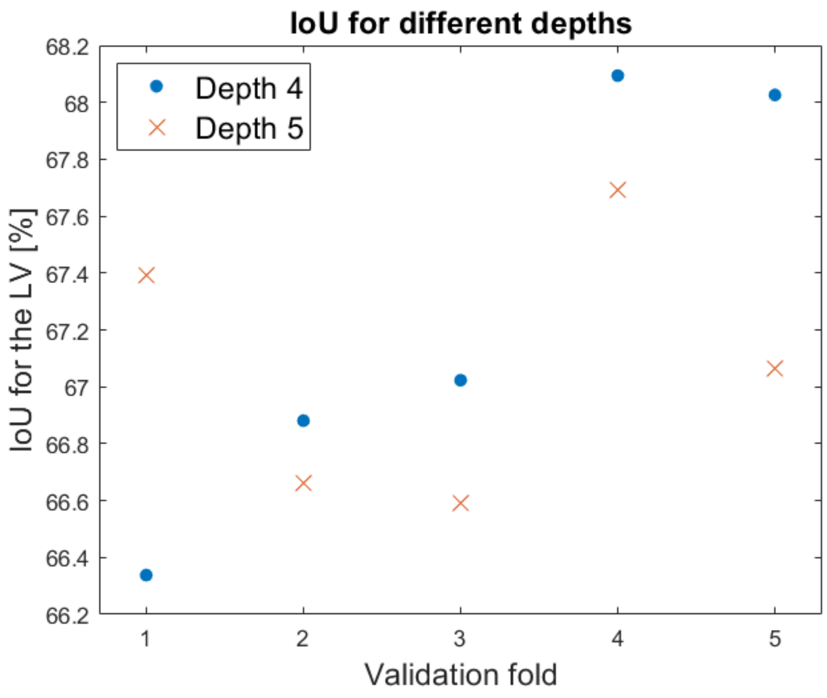

Figure 19: The resulting IoU when training using a U-net depth of 4 vs 5.

For the artificial class, the results with and without the class were compared. In addition, different settings for creating the artificial class were compared. Comparisons between using a square with side 10, a square with side 8 and a disk with diameter 10 for the dilation can be seen in Figure 20. The result for using a square of side 10 was not significantly better than using size 8 ($p = 0.30$). However, using the artificial class was significantly better than not using it ($p = 1.8 \cdot 10^{-6}$) and the square shape was significantly better than the disk of size 10 ($p = 7.3 \cdot 10^{-4}$).

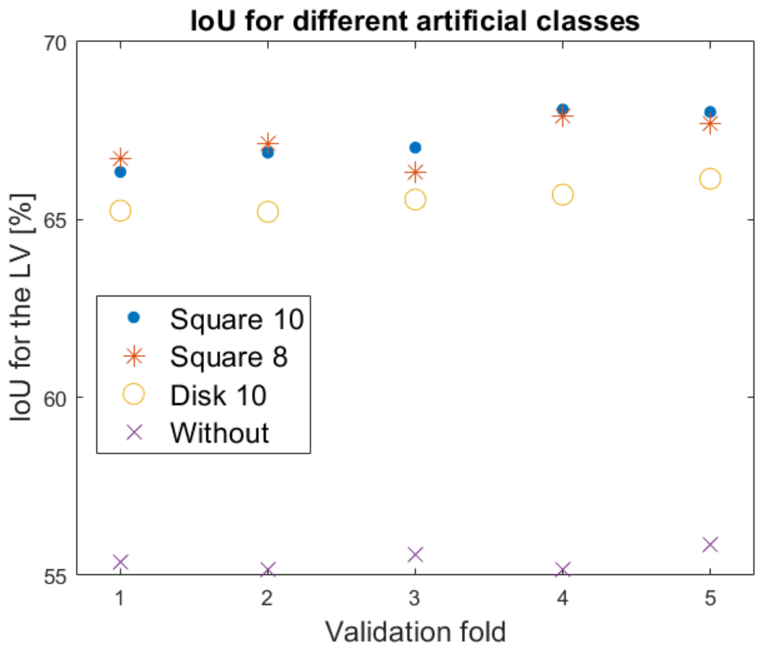

Figure 20: Comparison of the results using a square of side 10, a square of side 8 and a disk of diameter 10 for creating the edge class. The result without the artificial class is also shown.

The networks that were only trained on one of the three views at a time were compared with the network that was trained on all views. This can be seen in Figure 21. The network that was trained on all images was significantly better than any of the networks that were only trained on one type of image ($p = 0.006$ for the 2CH network, $p = 0.002$ for the 3CH network and $p = 0.007$ for the 4CH network). The results per view can be seen in Figure 22 and Figure 23. Note that the networks that were trained on only one type of view were still able to segment images from the other views with some success.

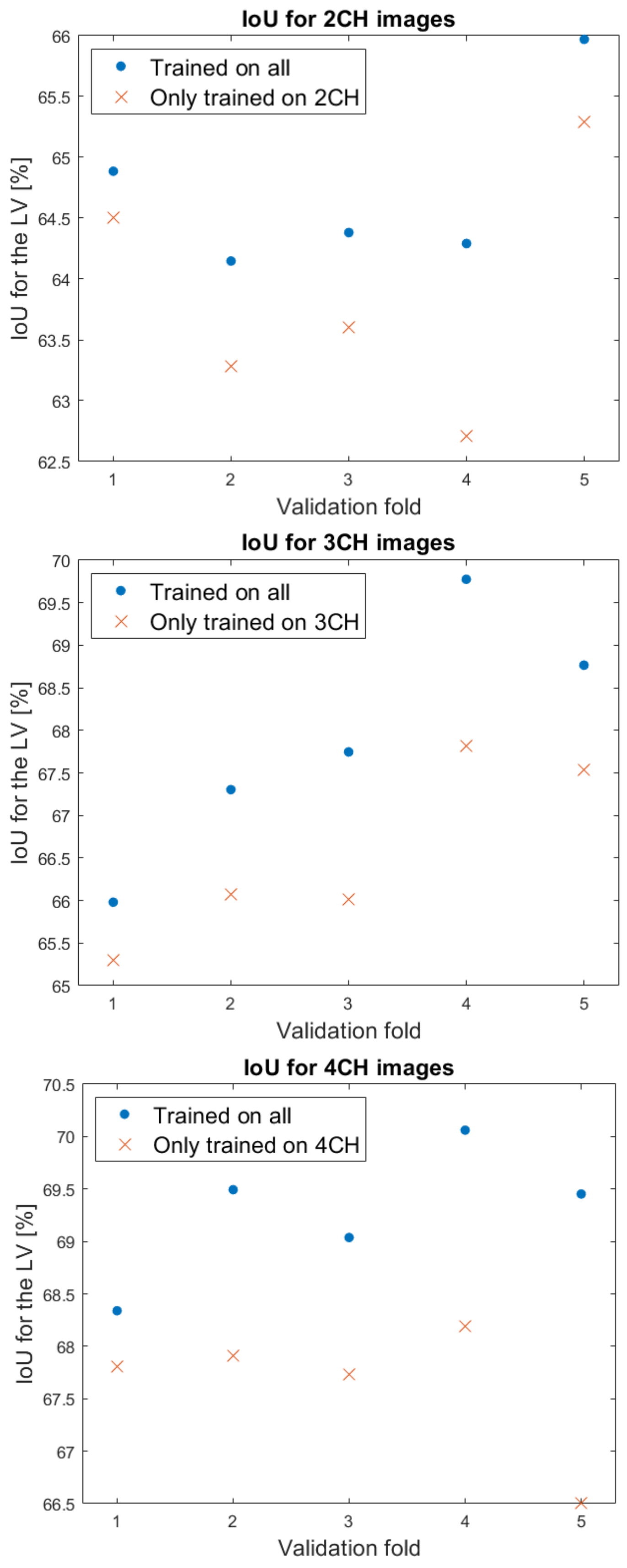

Figure 21: The IoU for the networks trained on one view compared to the network trained on all views. Top: 2CH, middle: 3CH, bottom: 4CH.

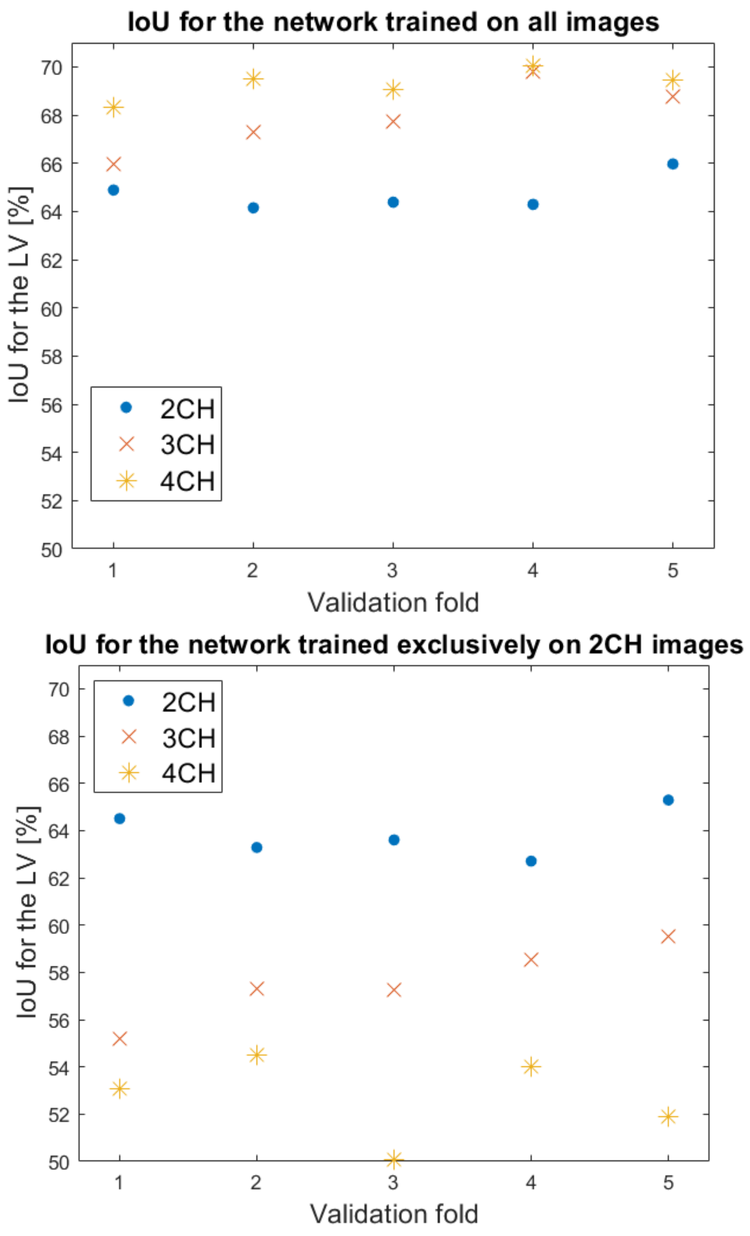

Figure 22: The IoU per view for the network trained on all views (top), only 2CH images (bottom).

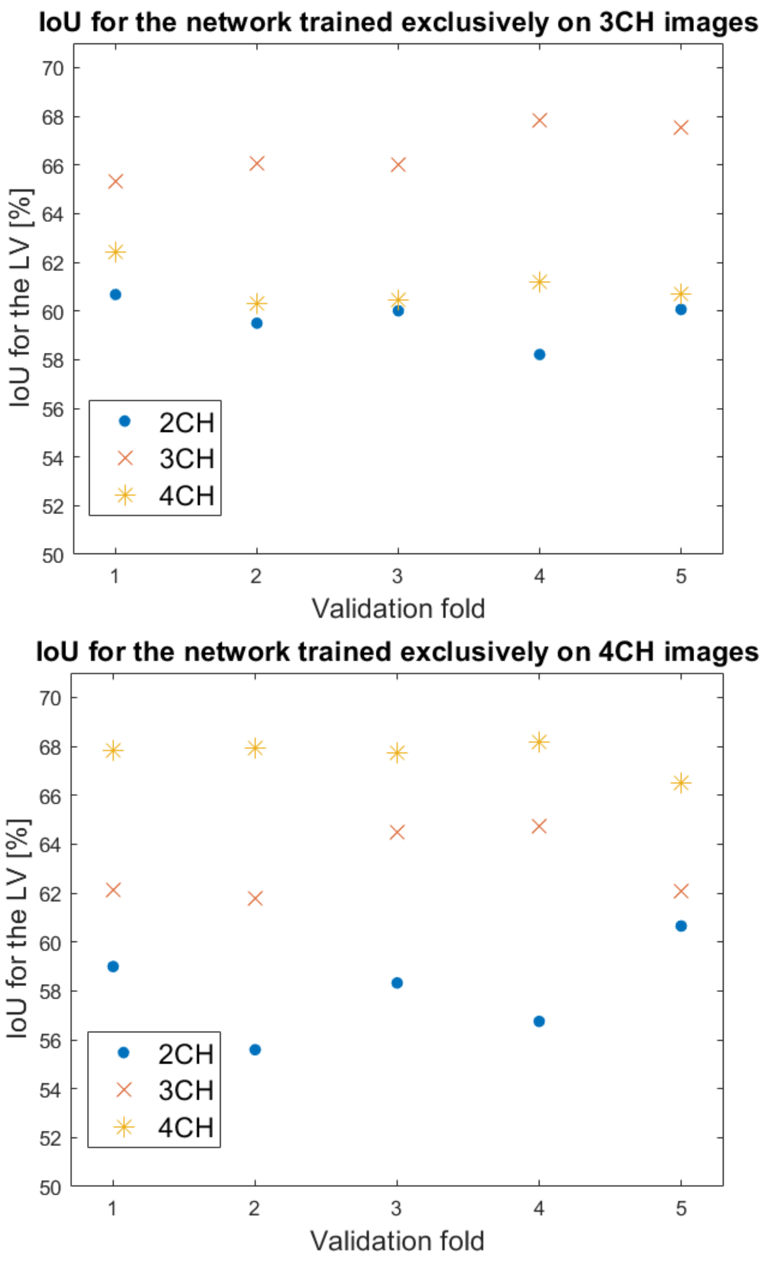

Figure 23: The IoU per view for the network trained on only 3CH images (top) and only 4CH images (bottom).

Augmentations

Additional augmentations were tested during the preliminary tests but were not part of the other models due to the poor results. One of the unused augmentations was to change the contrast of the images. The result always got worse when augmenting the contrast even when it was only applied moderately. Perhaps the contrast of the data is very similar between datasets due to the MR-settings being nearly the same across the datasets. As mentioned previously, using feature tracked segmentations on other time steps than the end-diastole time was tested. During preliminary testing, the IoU for the left ventricle decreased with about 14 percentage points compared to the best model. Therefore, due to both the poor results as well as complications with applying K-fold cross-validation to the feature tracked data, this was not tested using K-fold cross-validation or used in any of the other models.

The final models

Using the best settings for the networks as previously seen in Table 1 and Table 2, two networks were trained. One of the networks was compatible with only newer versions of MATLAB (2020a and above), since it normalized the input data to the network to have zero mean and standard deviation of 1, and the other network was compatible with 2019a since it only normalized the data to have zero mean. As seen in Figure 24, the standard deviation of the IoU for the 2019a compatible net was lower than for the 2020a compatible net and the mean was lower as well. Despite this, it cannot be concluded that the 2020a net is better than the 2019a net ($p = 0.085$).

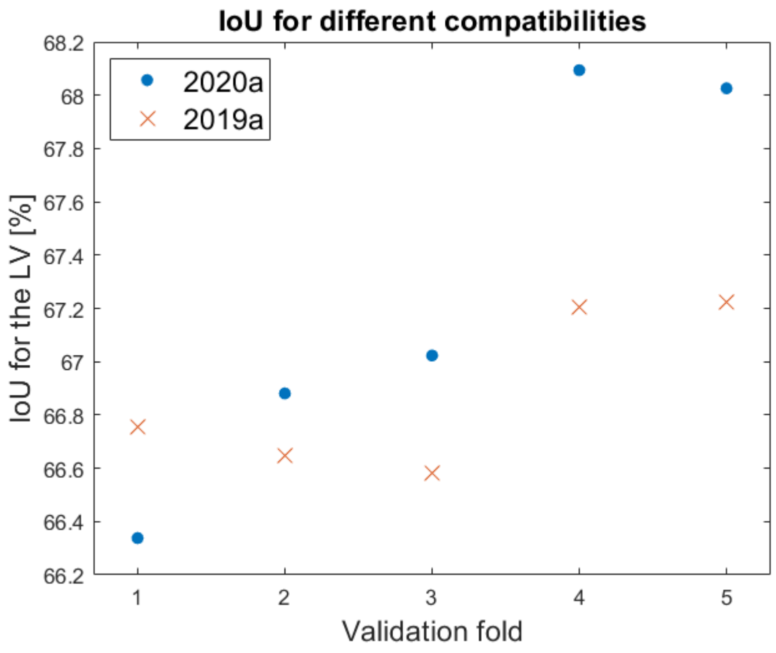

Figure 24: Comparing the IoU for the 2020a compatible network and the 2019a compatible network.

These two models were then trained on all five folds and tested on the test data. As seen in Table 3, the results on the test set were nearly the same for both networks (overall 68.4 % IoU for the 2020a compatible net vs. 68.3 % for the 2019a compatible net). When comparing the IoU for each of the dataset sources from the test set independently (Table 3) for the two final models, it is clear that some of the dataset sources were more difficult to segment than others. The CRT (Cardiac Resynchronization Therapy) set seems to be the most difficult while the CO (Cardiac Output) set seems to be the easiest.

Table 3: The IoU for each of the dataset sources for both the final 2020a compatible net and the 2019a compatible net.

Network	Total	AL	CHILL	CHILL-6-month	PAH	CO	CRT	ICD
2020a	68.4	65.8	71.6	66.5	67.6	73.3	60.9	67.5
2019a	68.3	65.2	71.3	66.0	67.1	73.5	60.3	68.0

Segmentation

An example image of a predicted segmentation compared with both a predicted segmentation without smoothing and closing as well as with a manual segmentation can be seen in Figure 25. The prediction is very similar to the manual delineation; however, it is slightly more jagged, especially without smoothing.

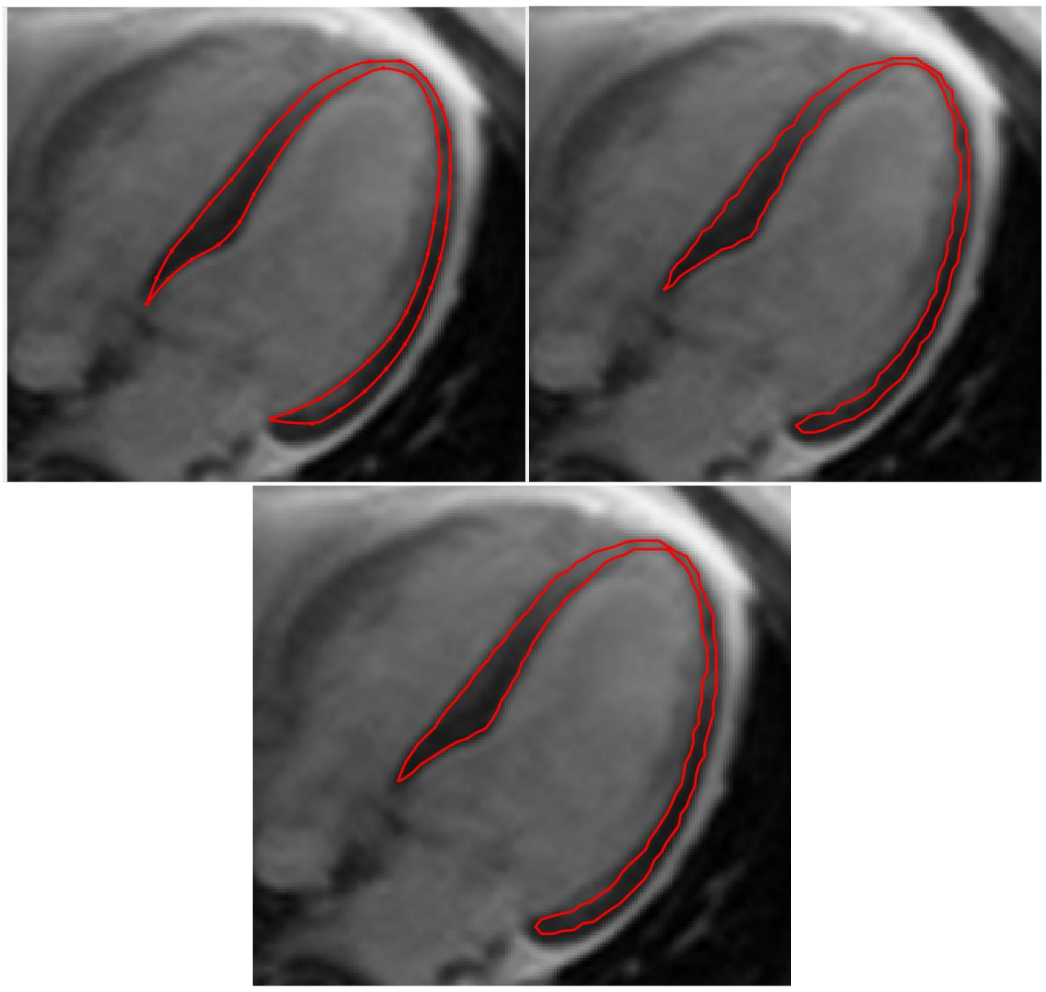

Figure 25: A manually delineated left ventricle (top left), a predicted delineation without smoothing (top right) and a predicted delineation using smoothing (bottom)

The model was tested on other timesteps than the end diastole time as well, but due to the lack of manual delineations, the results could only be assessed visually. Overall, the results on other timesteps were very good with some exceptions. An example of a good segmentation can be seen in Figure 26 which shows the end-systole time step for the same heart and view as in Figure 25 (no manual delineation exists to compare with). Sometimes the segmentations fail somewhat for a single or a couple of frames where either the segmentation is discontinuous, or an extra part of the heart has been included in the segmentation. An example of this can be seen in Figure 27 where, for a couple of frames, part of the valve has been segmented in addition to the myocardium.

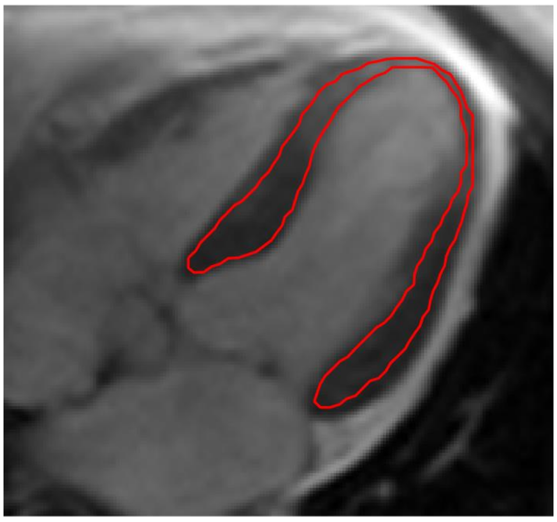
Figure 26: The predicted segmentation for the end-systole time step of the same heart as in Figure 25.

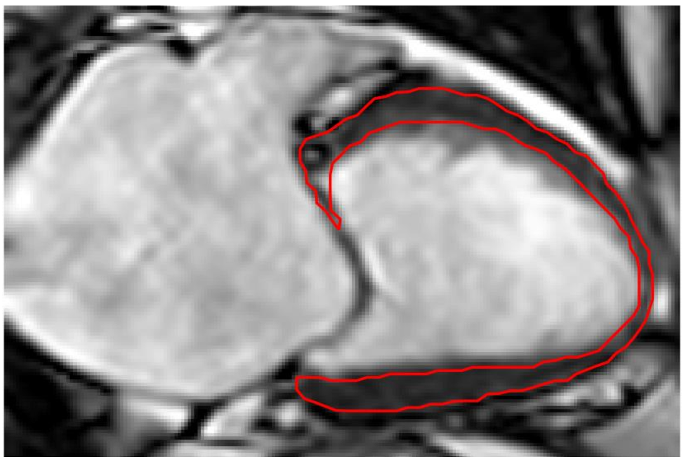
Figure 27: A predicted segmentation where part of the valve has been included in the delineation.

In addition, the model was tested on uncropped data. This was compared to the segmentations when using the automatic and manual cropping techniques. Since no manually delineated uncropped data was available, only manual inspections could be done. As seen in Figure 28, the uncropped segmentation is quite good but the segmentation using the manual and the automatic cropping methods were significantly better. The manually and automatically cropped images gave very similar segmentations with the most pronounced difference being how they were cropped.

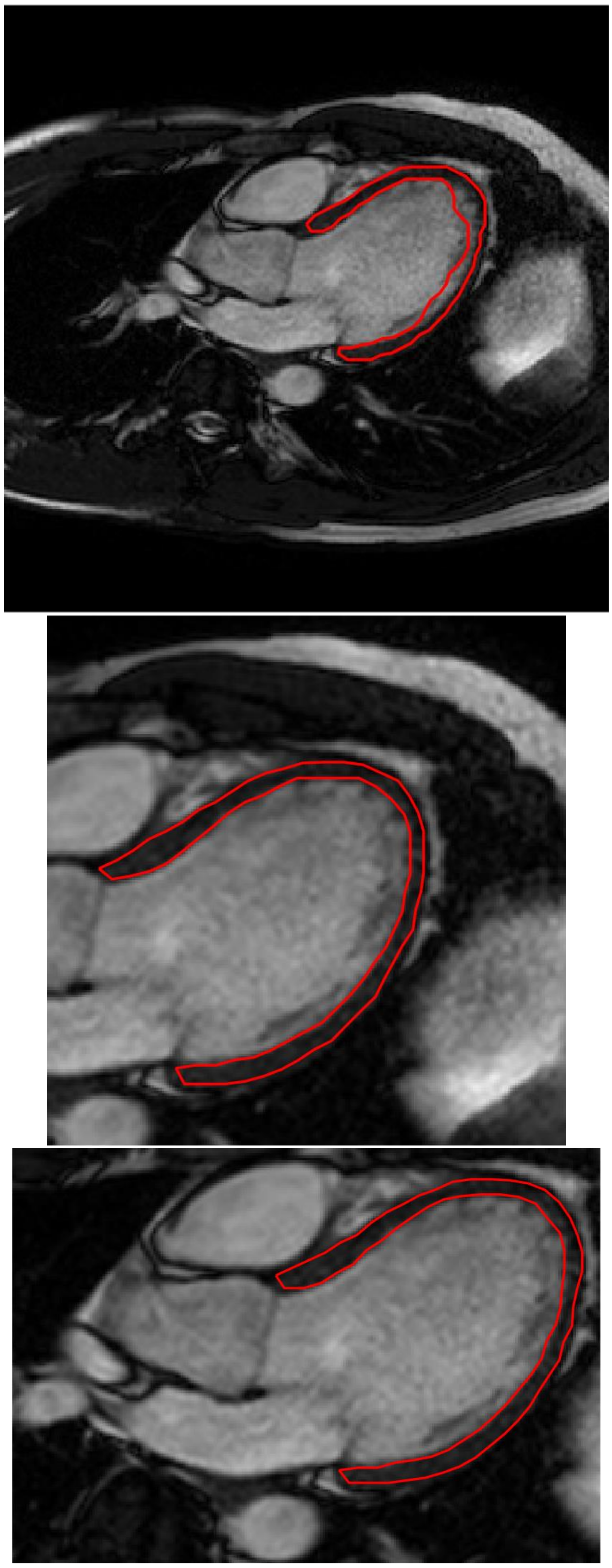
Figure 28: Automatic segmentation for an uncropped (top), automatically cropped (middle) and manually cropped (bottom) image.

Strain

For the final model it takes about 2 seconds to segment all 30 time frames from all three views when running on the GPU after loading the network. The first time the prediction is run, and the network has to be loaded, the segmentation takes about 6.5 seconds. After the network has been loaded, all subsequent predictions take 1-2 seconds depending on if all timesteps are segmented or if only the EDT is segmented. On a four-year-old mid-range laptop with a GPU, the computations take roughly 12 seconds regardless of how many frames are segmented and 27 seconds if the network needs to be loaded.

The two final models were used for strain calculations on all 101 test patients. The results were visualized in Bland Altman plots and can be seen in Figure 29 for the 2020a compatible network and Figure 30 for the 2019a compatible network. The results are similar but the 2020a compatible network performs slightly better with a coefficient of variation of 4.56 % compared to 5.06 % of the 2019a compatible network. However, the Pearson's correlation coefficient was 1.00 for both networks and the r^2-value was 0.99 for both networks.

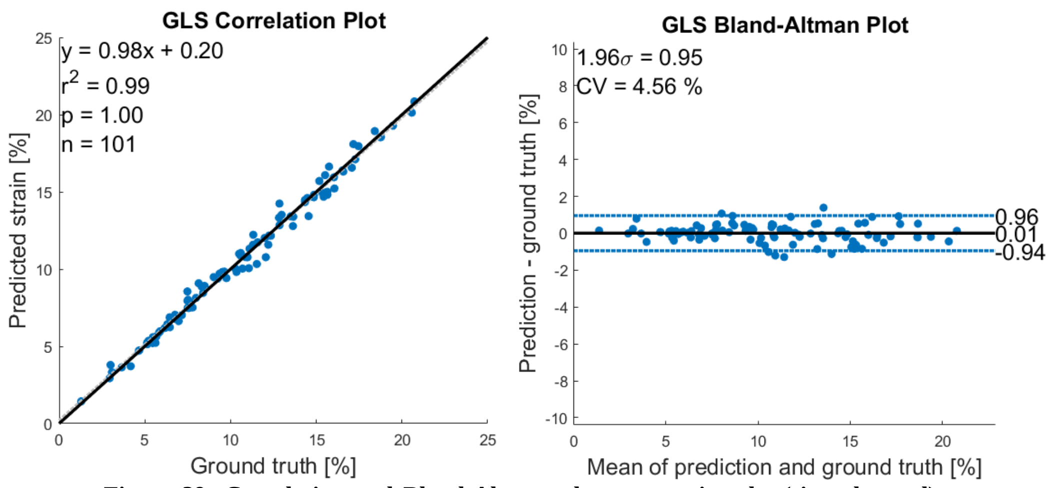

Figure 29: Correlation and Bland-Altman plots comparing the (sign changed) global longitudinal strain for the 2020a compatible network.

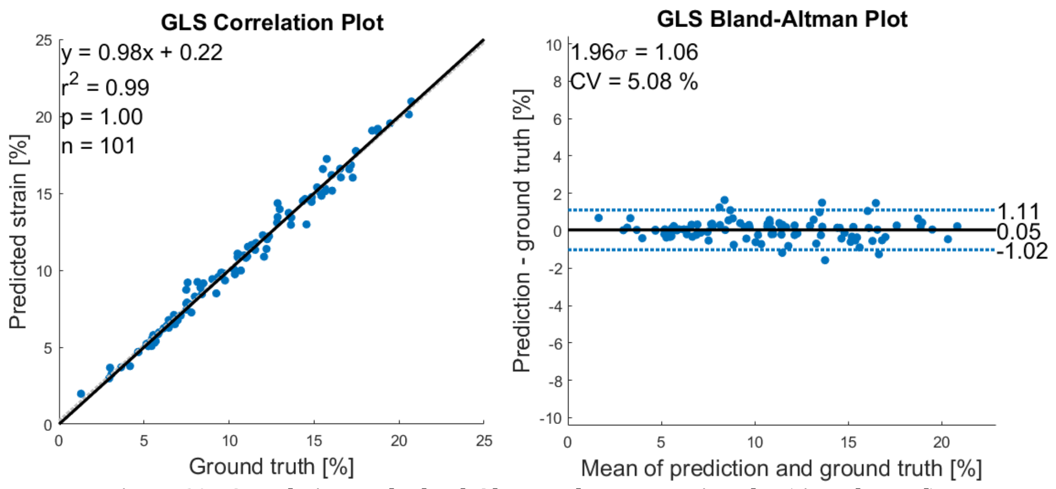

Figure 30: Correlation and Bland-Altman plots comparing the (sign changed) global longitudinal strain for the 2019a compatible network.

When calculating the strain without closing or smoothing, the results are very nearly the same as when using closing and smoothing. However, as discussed above, the segmentations that use smoothing are visually more appealing.

AV-plane displacement

The AV-plane displacement was only evaluated using the graphs seen in Figure 31 due to the lack of data to compare with. Excluding the noisy peaks, the displacement increases at the beginning of the heart cycle which is followed by a decrease. Some of the datasets had significantly worse graphs than those seen below with more noise. It should be noted that the displacement curves are expected to return to zero after one cardiac cycle. It was determined that significantly more work would be needed to achieve physiological displacement curves.

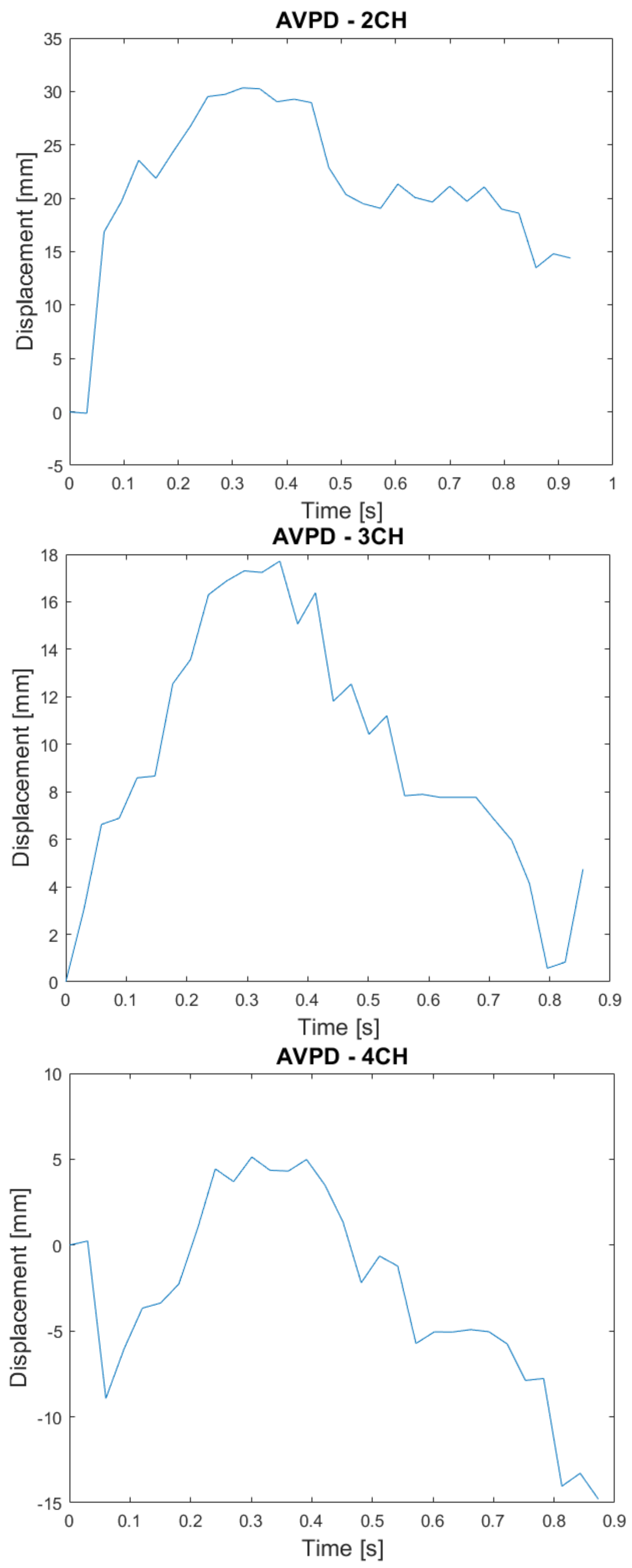

Figure 31: The AV-plane displacement over time for the 2CH (top), 3CH (middle) and 4CH (bottom) view of the same heart.

DISCUSSION

An automatic method for segmenting the myocardium of the left ventricle was created that could be used for accurate strain calculations. The segmentations from the network perform the best on cropped data. The cropping can be performed either manually or automatically.

Comparing dataset sources

As seen in Table 3, the IoU for the final model was around 67 % for four of the seven dataset sources while the result was higher for two of the sources and significantly lower for one of them. The dataset source with the best result was the cardiac output, or CO, set with an IoU of 73.3 %. This result was not entirely surprising since the CO-set was expected to contain the healthiest patients since it included both healthy volunteers and athletes in addition to sick patients. Most of the other set contained mainly sick patients. The CRT-set was the most difficult by far, with an IoU of only 60.9 %. Since the CRT-set contained patient in the need of cardiac resynchronization therapy, most, if not all, patients were very sick and suffered from moderate or severe heart failure. The hearts of these patients could therefore have a slightly different shape or size from healthier patients. In addition, some of these patients might have a reduced myocardium wall thickness which makes it more difficult to segment. However, since the results from the strain tests indicate that the strain is accurately calculated with a very low coefficient of variation regardless of dataset source, it is possible that the strain can be calculated accurately even with a lower IoU as long as most of the left ventricle is correctly segmented. For this reason, we expect the strain results to be good for all types of patients even when the segmentation is slightly worse.

Artificial edge-class

Since both the weighted cross-entropy loss as well as the Dice loss had problems with finding the edge of the left ventricle, with one loss overestimating the size regularly and the other underestimating it, a way to help the model segmenting the edge was needed. By introducing a new class surrounding the ventricle, incorrectly segmented pixels around the edge would be more heavily penalized since the artificial class was quite small. In this way, the model would be more likely to segment the edges correctly and therefore give as a better prediction of the entire left ventricle. The improvements that the artificial edge-class gave seemed to be consistent regardless of the loss function, augmentations, and which data it was applied to which suggests that it is a consistent improvement for left ventricle segmentations.

Loss functions

Different loss functions were also tested before settling on the weighted cross-entropy loss. The first loss function that was tested was the standard binary cross entropy error which was quickly abandoned due to the models tending to ignore the ventricle entirely. This was unsurprising since if the model were to classify everything as background, it would still get an accuracy of 95 %. The weighted cross-entropy error was therefore introduced to counter this problem which it also did quite well. Since the most important metric for evaluating the segmentations during training is the IoU or Dice scores, the Dice loss was tested as another loss function. For early tests when only half of the data was available, this was more successful than the weighted cross-entropy error. However, the Dice loss seemed to underestimate the size of the segmentations while the weighted cross-entropy overestimated it. This led to the introduction of the combo loss which was used for a while. When all data was available, the weighted cross-entropy turned out to be the optimal loss function. Perhaps the original data (i.e., from the AL, CHILL, CHILL-6-MONTH and PAH sources) contained a bias that suppressed the usability of the cross-entropy, or perhaps the larger dataset was more varied in a way that the weighted cross-entropy had more potential to learn.

Clinical limitations

When designing a model that should be usable in the clinic, there are several limitations that must be considered. For example, Segment currently supports MATLAB 2019a specifically, and the best model is only compatible with MATLAB 2020a and above. For this reason, the model that will be used clinically is a slightly altered version of the best model with a slightly worse performance (see Figure 24, Figure 29 and Figure 30). Another limitation that was considered was the size of the network. When training a network using a U-net depth of 5, the size is over 300 MB while using a depth of 4 leads to a network with a size of less than 100 MB. In addition to the smaller file size, the smaller network will also load significantly quicker. This led to the author preferring a smaller network over the larger. The mean for the smaller network was slightly higher even though the improvement was insignificant ($p = 0.30$). The reason why the difference was negligible could be that the image size has been reduced to such a degree that there is no more useful information to extract from the image. At depth 5, the image size will be 8x8 pixels which means that about 3 pixels will be of the class “left ventricle” since roughly 95 % of each image is background. If the image size were larger before being sent to the network, perhaps a depth of 5 would be beneficial.

An additional consideration regarding the file size was that one network being able to segment all three long-axis views was far more preferable than individual networks for each view due to the increased file size of having three networks. As seen in Figure 21, the results for using three individual networks versus just using one are similar. However, the network that was trained on all views performed better on each individual view than the corresponding networks that were trained on only one view. This is probably due to the actual myocardium having a very similar appearance over all views with the main difference being the background. Due to the better result as well as the file size consideration, it was decided to use only one network for all views.

The final network sometimes fails to segment the left ventricle correctly, especially when segmenting all time frames. If segmentations for all time frames are needed, this could be a potential problem. However, the current usage of the model will be limited to the strain calculations and since only one frame is needed for those calculations, this will not be a problem. If the model is to have other uses in the future, it is possible that some improvements would be needed for segmentations of all time frames to be accurate.

Strain

The model was successfully used to estimate the global longitudinal strain of the left ventricle. The coefficient of variation was 4.56 % for the best model (Figure 29) which is quite small. This can be compared to the inter- and intra-observer coefficients of variation which were found to be 4.0 % and 5.0 % respectively (curiously the intra-observer variation was larger than the inter-observer variation) using Segment according to Barreiro-Pérez et al. [10]. Since the coefficient of variation between the automatic model and the ground truth, which was delineated by two different persons, is of roughly the same size as the inter-observer variability, the calculated results could be considered as valid and useful. Since the segmentation is easy to visually assess, it is very simple to check that the automatic segmentation hasn't failed by only segmenting half of the ventricle or similar before trusting the strain results blindly.

Other tests

Due to time constraints and lack of data, the results for the AV-plane displacement are very limited and it is difficult to assess the results. However, it was possible to find the AV-points of the heart as well as finding a suitable reference point that can be used over all time frames. In addition, the displacement between the AV-plane and the reference point could be calculated. It is unknown if the calculations can be used at all or if the estimated displacement is too rough to be of any real clinical use. However, since the AVPD was able to be calculated, it is possible that with more time, a better algorithm for calculating the AVPD can be created that is more useful clinically. Currently, the largest known problem is that the segmentation flickers between time steps due to the predicted segmentations being independent, and that the AV-plane point is not returning to its original position. This could potentially be solved by smoothing between the time steps or training a better and more stable model. Some minor work was done on temporally smoothing the images but not enough for any conclusive results. As of right now, it is unlikely that the AVPD will be useful from the segmentations due to the high amount of noise.

The results from the cropping tests strongly indicate that cropping is necessary. Cropping is currently done manually and could still be done manually with the automatic model. However, for the model to be entirely automatic, an automatic cropping algorithm would be advantageous. The current automatic cropping method seems to work even though it has only seen very limited testing. By evaluating the automatic cropping algorithm further and discussing with clinicians, it could be tweaked to crop the image to the liking of the clinicians and thereafter be used with the strain calculations and segmentations.

Sources of errors

A potential problem with the model is the limitations within the datasets that were used. An advantage is that the datasets were delineated by more than one physician; however, since the data was only delineated by two physicians, biases related to the way they segment could limit the model. In addition, the delineations had previously been used in research and were not originally made for training deep learning models. Due to this, some delineations could be more exact such as the one seen in Figure 32.

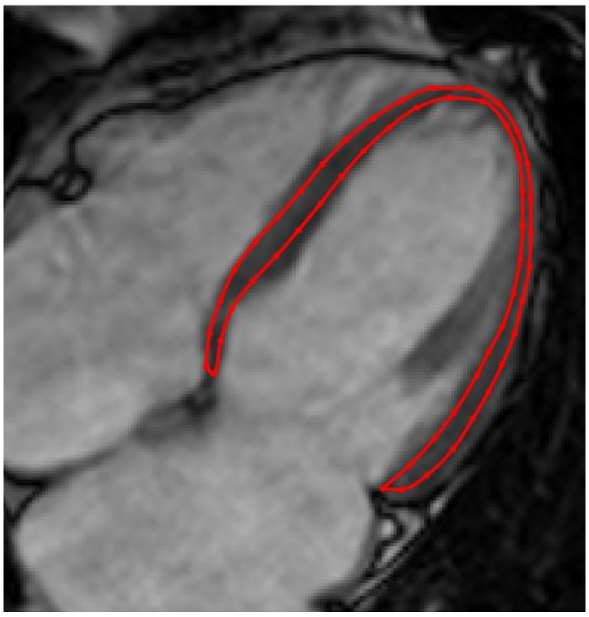
Figure 32: An example of a manually delineated image. Note that the bottom right part of the segmentation doesn't follow contour of the myocardium.

Future work

One “simple” way to potentially improve the model would be to improve the data that was used. Since the data was not originally delineated to be used for machine learning, some delineations could be made more exact such as previously seen in Figure 32. If more exact data were used for training, a better model could be created and if more exact data were used for validation and testing, the likelihood of choosing the best model would also increase. If an expert went through all images and either made the segmentations more exact or excluded the less exact images, the data would be more suitable for machine learning. However, due the large number of images, roughly 1800, it would be a considerable time investment for the expert which might not be worth the effort due to the already great results.

In addition to improving the existing data, using more data from other scanners, patients and maybe even other animals, could improve the model. Unsegmented data could also be of use to assess the performance of the model in other conditions, such as with patients with other diseases or on animals.

Another improvement would be to develop the automatic cropping algorithm further. This could be done in a couple of ways. One way would be to discussing how the clinicians want the images to be cropped and try to optimize the algorithm to crop in this way. Another way would be to measure and test on more uncropped images to find the appropriate buffer sizes around the segmentation and other settings. Finally, manually delineated uncropped images could be used for testing the cropping.

The clinicians will probably feel inclined to adjust the predicted segmentations before trusting the calculated strain since they know that a model has segmented the ventricle instead of a human. To ensure that this is necessary, it would be interesting to compare the calculated strain before and after such adjustments as well as comparing it with the inter-observer variation. Since the coefficient of variation between the prediction and ground truth was found to be of the same order of magnitude as the inter-observer variability, it is very doubtful that any tweaking of the model would be necessary in most cases. However, if the segmentation was discontinuous or if part of the valve is segmented as in Figure 27, manual tweaking is recommended. Additionally, it could be evaluated how much the strain changes when the clinician is given a manual delineation made by someone else to tweak to compare with the changes when tweaking predictions from the model.

The algorithm for calculating the AV-plane displacement has not been tested yet. If the AVPD is going to be used in conjunction with the predicted segmentations, the algorithm needs to be tested as well as developed further. A method of skipping frames with poor segmentation or a way of smoothing between time frames is probably needed to get a better result. Due to the currently mediocre results, it is doubtful that it would be worth the time-investment of developing it further. An alternative method would be to use a dedicated residual network to track the AV-plane points.

An algorithm for calculating the volume in the left ventricle could be created using the predicted segmentations. Since most segmentations are very accurate and since minor differences in the contour will not affect the area and volume inside the left ventricle too much, it is reasonable to assume that the algorithm could become reasonably accurate. However, when using an algorithm to estimate the volume, the user needs to be mindful of discontinuous segmentations since this would make the area calculations impossible. The algorithm should therefore be able to exclude or warn the user about incomplete segmentations and advise the user to manually delineate the missing frame(s).

CONCLUSION

In conclusion, the model was able to segment the myocardium of the left ventricle of 2-chamber, 3-chamber as well as 4-chamber cardiac MRI images in all time frames. The predicted segmentations can be used for calculation of the global longitudinal strain with a coefficient of variation of 4.56 % when compared to the ground truth. However, for the best results, the images must be cropped. This can be done either automatically or manually.

REFERENCES

[1] World Health Organization, “Cardiovascular diseases (CVDs),” 17 May 2017. [Online]. Available: https://www.who.int/en/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds). [Accessed 20 April 2021].

[2] O. Ronneberger, P. Fischer and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention*, 2015.

[3] G. Roth, D. Abate, K. H. Abate, S. Abay, N. Abbasi, H. Abbastabar, L. Abd-Allah, J. Abdela, J. Ärnlöv and C. J. L. Murray, “Global, regional, and national age-sex-specific mortality for 282 causes of death in 195 countries and territories, 1980-2017: a systematic analysis for the Global Burden of Disease Study 2017,” *The Lancet Global Health,* vol. 392, no. 10159, pp. 1736-1788, 2018.

[4] P. A. Rinck, “A short history of magnetic resonance imaging,” *Spectroscopy Europe,* vol. 20, no. 1, pp. 7-9, 2008.

[5] J. J. Park, J.-B. Park, J.-H. Park and G.-Y. Cho, “Global Longitudinal Strain to Predict Mortality in Patients With Acute Heart Failure,” *Journal of the American College of Cardiology,* vol. 71, no. 18, pp. 1947-1957, 8 May 2018.

[6] R. Willenheimer, C. Cline, L. Erhardt and B. Israelsson, “Left ventricular atrioventricular plane displacement: An echocardiographic technique for rapid assessment of prognosis in heart failure,” *Heart,* vol. 78, no. 3, pp. 230-236, 1997.

[7] G. M, L. MS and L. D, “Echocardiographic determinants of clinical outcome in subjects with coronary artery disease (the Framingham Heart Study),” *The American journal of cardiology,* vol. 70, no. 11, pp. 971-976, 15 October 1992.

[8] F. H. Sheehan, R. Doerr, W. G. Schmidt, E. L. Bolson, R. Uebis, R. von Essen, S. Effert and H. T. Dodge, “Early recovery of left ventricular function after thrombolytic therapy for acute myocardial infarction: an important determinant of survival,” *Journal of the American College of Cardiology,* vol. 12, no. 2, pp. 289-300, 1988.

[9] E. Heiberg, J. Sjögren, M. Ugander, M. Carlsson, H. Engblom and H. Arheden, “Design and validation of Segment - freely available software for cardiovascular image analysis,” *BMC Medical Imaging,* vol. 10, pp. 1-13, 2010.

[10] M. Barreiro-Pérez, D. Curione, R. Symons, P. Claus, J.-U. Voigt and J. Bogaert, “Left ventricular global myocardial strain assessment comparing the

reproducibility of four commercially available CMR-feature tracking algorithms.," *European Radiology,* vol. 28, no. 12, pp. 5137-5147, 1 December 2018.

[11] J. Ba and D. P. Kingma, "Adam: A Method for Stochastic Optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015.

[12] J. Bogaert, S. Dymarkowski, A. Taylor and V. Muthurangu, Clinical Cardiac MRI, 2nd ed., Berlin Heidelberg: Springer, 2012.

[13] R. M. Lang, L. P. Badano, V. Mor-Avi, J. Afilalo, A. Armstrong, L. Ernande, F. A. Flachskampf, E. Foster, S. A. Goldstein, T. Kuznetsova, P. Lancellotti, D. Muraru, M. H. Picard, E. R. Rietzschel, L. Rudski, K. T. Spencer, W. Tsang and J.-U. Voigt, "Recommendations for cardiac chamber quantification by echocardiography in adults: an update from the American Society of Echocardiography and the European Association of Cardiovascular Imaging," *Journal of the American Society of Echocardiography*, vol. 28, no. 1, pp. 1-39, 2015.

[14] F. Seemann, U. Pahlm, K. Steding Ehrenborg, E. Ostenfeld, J.-L. Dubois-Randé, D. Atar, H. Arheden, M. Carlsson and E. Heiberg, "Time-resolved tracking of the atrioventricular plane displacement in Cardiovascular Magnetic Resonance (CMR) images," *BMC Medical Imaging,* vol. 17, no. 19, 2017.

[15] M. D. Cerqueira, N. J. Weissman, V. Dilsizian, A. K. Jacobs, S. Kaul, W. K. Laskey, D. J. Pennell, J. A. Rumberger, T. J. Ryan and M. S. Verani, "Standardized Myocardial Segmentation and Nomenclature for Tomographic Imaging of the Heart," *Journal of Cardiovascular Magnetic Resonance,* vol. 4, no. 2, p. 203, 2002.

[16] N. Beitner, J. Jenner and P. Sörensson, "Comparison of Left Ventricular Volumes Measured by 3DE, SPECT and CMR," *Journal of cardiovascular imaging,* vol. 27, no. 3, pp. 200-211, 2019.

[17] R. Nash and K. O'Shea, "An Introduction to Convolutional Neural Networks," 2015.

[18] I. Goodfellow, Y. Bengio and A. Courville, "Deep Learning," MIT Press, 2016.

[19] D. J. Livingstone, J. Zou, Y. Han, S.-S. So and J. M. Walker, Overview of Artificial Neural Networks, Totowa: Humana Press, 2009.

[20] G. Csurka, D. Larlus and F. Peronnin, "What is a good evaluation measure for semantic segmentation?," in *Proceedings of the British Machine Vision Conference*, 2013.

[21] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

[22] R. Durall, M. Keuper and J. Keuper, “Watch Your Up-Convolution: CNN Based Generative Deep Neural Networks Are Failing to Reproduce Spectral Distributions,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, 2020.

[23] S. Jadon, “A survey of loss functions for semantic segmentation,” in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, 2020.

[24] S. A. Taghanaki, Y. Zheng, S. Kevin Zhou, B. Georgescu, P. Sharma, D. Xu, D. Comaniciu and G. Hamarneh, “Combo Loss: Handling Input and Output Imbalance in Multi-Organ Segmentation,” *Computerizeed Medical Imaging and Graphics*, vol. 75, pp. 24-33, July 2019.

[25] L. Yang and A. Shami, “On hyperparameter optimization of machine learning algorithms: Theory and practice,” *Neurocomputing,* no. 415, pp. 295-316, 20 November 2020.

[26] L. Zhang, X. Wang, D. Yang, T. Sanford, S. Harmon, B. Turkbey, H. Roth, A. Myronenko, D. Xu and Z. Xu, “When Unseen Domain Generalization is Unnecessary? Rethinking Data Augmentation,” *Computing Research Repository,* vol. abs/1906.03347, 12 Jun 2019.

[27] A. Luque, A. Carrasco, A. Martín and A. de las Heras, “The impact of class imbalance in classification performance metrics based on the binary confusion matrix,” *Pattern Recognition,* vol. 91, pp. 216-231, 2019.

[28] T. Eelbode, J. Bertels, M. Berman, D. Vandermeulen, F. Maes, R. Bisschops and M. Blaschko, “Optimization for Medical Image Segmentation: Theory and Practice When Evaluating With Dice Score or Jaccard Index,” *IEEE Transactions on Medical Imaging,* vol. 39, no. 11, pp. 3679-3690, 2020.

[29] K. Bratis, A. Lindholm, R. Hesselstrand, H. Arheden, G. Karabela, E. Stavropoulos, G. Katsifis, G. Kolovou, G. D. Kitas, P. P. Sfikakis, L. Koutsogeorgopoulou, S. Mavrogeni and E. Ostenfeld, “CMR feature tracking in cardiac asymptomatic systemic sclerosis : Clinical implications,” *PLoS ONE,* vol. 14, no. 8, pp. 1-13, 2019.

[30] D. Erlinge, M. Gotberg, I. Lang, M. Holzer, M. Noc, P. Clemmensen, U. Jensen, B. Metzler, S. K. James, H. E. Botker, E. Omerovic, H. Engblom, M. Carlsson, H. Arheden, O. Östlund, L. Wallentin, J. Harnek and G. K. Olivecrona, “Rapid Endovascular Catheter Core Cooling Combined With Cold Saline as an Adjunct to Percutaneous Coronary Intervention for the

Treatment of Acute Myocardial Infarction: The CHILL-MI Trial," *Journal of the American College of Cardiology,* vol. 63, no. 18, pp. 1857-1865, 2014.

[31] E. AW, L. A, J. R, I. A, S. GJ, W. J, R. G, R. A, M. C and Ostenfeld E, "Right ventricular function parameters in pulmonary hypertension: echocardiography vs. cardiac magnetic resonance," *BMC cardiovascular disorders,* vol. 20, no. 1, p. 259, 1 June 2020.

[32] A. Lindholm, R. Hesselstrand, G. Rådegran, H. Arheden and E. Ostenfeld, "Decreased biventricular longitudinal strain in patients with systemic sclerosis is mainly caused by pulmonary hypertension and not by systemic sclerosis per se," *Clinical Physiology and Functional Imaging,* vol. 39, no. 3, pp. 215-225, 2019.

[33] M. Carlsson, R. Andersson, K. M. Bloch, K. Steding-Ehrenborg, H. Mosén, F. Ståhlberg, B. Ekmehag and H. Arheden, "Cardiac output and cardiac index measured with cardiovascular magnetic resonance in healthy subjects, elite athletes and patients with congestive heart failure," *Journal of Cardiovascular Magnetic Resonance,* vol. 14, no. 1, pp. 51-57, 2012.

[34] R. Borgquist, M. Carlsson, H. Markstad, A. Werther-Evaldsson, E. Ostenfeld, A. Roijer and Z. Bakos, "Cardiac Resynchronization Therapy Guided by Echocardiography, MRI, and CT Imaging: A Randomized Controlled Study," *JACC: Clinical Electrophysiolog,* vol. 6, no. 10, pp. 1300-1309, 2020.

[35] R. Jablonowski, U. Chaudhry, J. Van Der Pals, H. Engblom, H. Arheden, E. Heiberg, K. C. Wu, R. Borgquist and M. Carlsson, "Cardiovascular Magnetic Resonance to Predict Appropriate Implantable Cardioverter Defibrillator Therapy in Ischemic and Nonischemic Cardiomyopathy Patients Using Late Gadolinium Enhancement Border Zone: Comparison of Four Analysis Methods," *Circulation: Cardiovascular Imaging,* vol. 10, no. 9, 2017.

[36] MATLAB, version 9.10.0 (R2021a), Natick, Massachusetts: The MathWorks Inc., 2021.

[37] J. Åkesson, "Automated bone segmentation in computed tomography using deep learning with distance maps," Lund, 2020.

[38] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen and K. H. Maier-Hein, "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation," *Nature Methods,* vol. 18, pp. 203-211, 2021.

[39] D. Giavarina, "Understanding Bland Altman analysis," *Biochemia medica,* vol. 25, no. 2, pp. 141-151, 5 June 2015.

Master's Theses in Mathematical Sciences 2021:E55
ISSN 1404-6342

LUTFMA-3457-2021

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

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