Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Trym Nordal

Automatic Detection of Mitral Annular Plane Systolic Excursion from Transesophageal Echocardiography Using Deep Learning

Master's thesis in Cybernetics and Robotics
Supervisor: Lasse Løvstakken
June 2019

NTNU
Norwegian University of Science and Technology

Trym Nordal

Automatic Detection of Mitral Annular Plane Systolic Excursion from Transesophageal Echocardiography Using Deep Learning

Master's thesis in Cybernetics and Robotics
Supervisor: Lasse Løvstakken
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Preface

This Master's thesis presents a pipeline for fully automatic detection of MAPSE in TEE B-mode recordings of the left ventricle. The task of automatic MAPSE detection from TEE data was proposed as a research topic for a Master's thesis by the Operating Room of the Future from St. Olavs University hospital in the spring of 2018. In the autumn of 2018, a preliminary project was conducted as preparation for the Master's project and the work presented in this thesis is a continuation of the preliminary project. However, the methods and results from the preliminary project could not be reused for this study.

The ultrasound data used in the current study was acquired by cardiologists at the Echocardiography Unit in the Clinic of Cardiology at St. Olavs University hospital. Export and scan-conversion has been performed by cardiologist Erik Andreas Rye Berg and NTNU researcher Gabriel Kiss. Data annotation has been done by me and fellow Master's student Torjus Haukom. The Operating Room of the Future provided access to the online GPU service *Floydhub* for tasks related to deep learning. The rest of the work presented in this thesis is done by the author, with guidance from Gabriel Kiss and Erik Andreas Rye Berg.

Acknowledgement

Working on this project has been very rewarding and exiting. I would like to express my gratitude to The Operating Room of the Future (FOR) at St. Olavs University hospital (Trondheim, Norway) for providing me with the opportunity to work on such an interesting project and for supporting me with all the necessary resources. I would like to thank my supervisor Lasse Løvstakken for enabling me to be a part of this project. I would also like to thank my family and friends for their love and support.

In addition, three people deserve to be specifically mentioned. I would like to express my sincerest gratitude to:

- **Gabriel Kiss**, my supervisor from The Operating Room of the Future, for great guidance and constructive feedback during the whole project.
- **Erik Andreas Rye Berg**, PhD candidate at the Department for Circulation and Medical Imaging at NTNU, for sharing expertise in the field of echocardiography with me.
- **Gina Fossum Bøen**, for continuous love and support, and for proofreading.

Abstract

Perioperative cardiac monitoring of patients undergoing surgery is vital in ensuring that the heart restores desired functionality. As of today, perioperative cardiac monitoring is performed manually, based on vital signs and clinical observations. Complete cardiac monitoring and assessment is reserved for major cardiac interventions, with transesophageal echocardiography (TEE) as a widely accepted and essential monitoring tool. Fully automatic and quantitative assessment of cardiac function during surgery can eliminate intra-observer variability, speed up the process, and make complete cardiac monitoring feasible for any type of surgery. This may in turn contribute to reduced risk of cardiovascular complications, which is currently one of the most common causes of deaths in the operating room.

In recent years, systems utilizing deep learning have revolutionized how complex tasks such as image segmentation and object detection are solved by computers, with successful applications in the field of medical imaging.

The study presented in this thesis is based on the hypothesis that perioperative monitoring and assessment of cardiac function from TEE can be automated by utilizing the power of deep learning. A pipeline for automatic detection of the global systolic functional parameter MAPSE (mitral annular plane systolic excursion) has been proposed, implemented and tested against clinical measurements from St. Olavs University hospital, Trondheim, Norway. The pipeline consists of a convolutional neural network (CNN) for detection of a set of mitral landmarks in two-dimensional TEE B-mode recordings of the left ventricle, and necessary post processing components in order to get a final MAPSE estimate. The CNN landmark detector has been trained with 131 two-chamber and four-chamber recordings.

When compared with clinically obtained MAPSE values from test data on 46 TEE recordings, the mean error of the proposed method is -0.08 ± 1.38 mm. The method does not yield significant systematic errors, but some outliers are produced, particularly in noisy recordings. Based on the results from this study, an abstract has been submitted to the conference EuroEcho19. The abstract is included in appendix A.

Table of Contents

List of Figures

List of Tables

Abbreviations

2C	:	Two-chamber
4C	:	Four-chamber
Adam	:	Adaptive moment estimation
AdaGrad	:	Adaptive gradient algorithm
ALAX	:	Apical long axis
ANN	:	Artificial neural network
AVPD	:	Atrioventricular plane displacement
CNN	:	Convolutional neural network
ED	:	End-systole
ECG	:	Electrocardiography
ES	:	End-systole
LV	:	Left ventricle
LVEF	:	Left ventricular ejection fraction
MAE	:	Mean Absolute Error
MAPSE	:	Mitral annular plane systolic excursion
MCC	:	Matthews Correlation Coefficient
MSE	:	Mean Squared Error
ReLU	:	Rectified linear unit
RMSProp	:	Root mean squared propagation
RNN	:	Recurrent neural network
SGD	:	Stochastic gradient descent
STE	:	Speckle tracking echocardiography
TDI	:	Tissue Doppler imaging
TEE	:	Transesophageal echocardiography
TNR	:	True negative rate
TPR	:	True positive rate
TTE	:	Transthoracic echocardiography

1 Introduction

1.1 Background

It is estimated that over 300 million surgeries are performed annually worldwide [1]. Major surgeries and interventions can have a significant negative impact on cardiac function. Perioperative cardiovascular complications are also some of the most common factors leading to an increased risk of death in the operating room [2]. The hearts of the patients undergoing comprehensive interventions are therefore monitored carefully before, during and after the procedures. As of today, perioperative cardiac monitoring in non-cardiac surgeries is generally based on vital signs and clinical observations. Complete monitoring of left ventricular function, is reserved for major cardiac interventions such as bypass surgery, vascular surgery and valve related interventions [3]. This procedure involves measuring blood pressure, heart rate, respiratory rate, blood oxygen saturation, performing hemodynamic monitoring, clinical observation and echocardiographic evaluation, which is based on ultrasound imaging of the heart [4]. Complete perioperative monitoring of cardiac function is a complex, time-consuming and resource heavy process.

Diagnostic ultrasound imaging is one of the most widely used imaging modalities in modern medicine [3]. Technological advancements have made it possible to visualize complex anatomical structures and to accurately estimate blood flow and tissue movements. Echocardiography has become an indispensable part of assessing and monitoring cardiac function [5, 6]. Transthoracic echocardiography (TTE) is the most prevalent type of echocardiography used to assess cardiac function in routine controls. In the operating room, however, transesophageal echocardiography (TEE) is more commonly used because the ultrasound probe is placed in the patient's esophagus and can stay there during the procedure. Both TTE and TEE are used to assess left ventricular systolic and diastolic function. By analyzing detailed images of the left ventricle, cardiac function parameters such as ejection fraction, strain and mitral annular plane systolic excursion (MAPSE) can be determined. These parameters are essential, providing important information on the

functionality of the heart.

Despite recent technological advancements, fully automatic and accurate methods for perioperative cardiac assessment are still in the research stage, and not yet integrated in clinical systems. Monitoring cardiac function automatically in the operating room can potentially provide two major advantages:

1. **Faster and more reliable results.**
   Manual measurements of cardiac function parameters are susceptible to intra- and inter-observer variability and are inherently time-consuming. A system performing automatic cardiac function assessment can eliminate variability and perform calculations much faster compared with manual assessment.

2. **Automatic cardiac monitoring for any type of surgery.**
   By doing perioperative assessment of cardiac function automatically it is much more feasible to include this in any type of surgery. This can potentially reduce the risk of cardiovascular complications by detecting earlier changes in the function of the heart.

Thus, the benefits of fully automatic and accurate cardiac function assessment in the operating room are significant.

In recent years, a new family of methods for machine learning has revolutionized fields such as computer vision, speech recognition and natural language processing. The common denominator for these machine learning methods is that they utilize layered structures of artificial neurons in order to learn feature representation of the input data with different abstraction levels. The process of training these multilayered networks to learn the feature representations of the input data is referred to as *deep learning*. Tasks such as image classification, image segmentation, object localization and landmark detection are some of the tasks within computer vision that has been revolutionized by deep learning [7]. Deep learning has also been successfully applied to medical imaging and there is an increasing amount of research on new applications of deep learning within the field of medicine.

The study presented in this thesis is based on the hypothesis that MAPSE can be automatically detected in TEE recordings of the left ventricle using deep learning. The proposed method uses a convolutional neural network, inspired by state-of-the-art methods, for mitral landmark detection and implements the necessary post processing steps for fully automatic derivation of MAPSE. The proposed pipeline provides an indication of the feasibility of fully automatic MAPSE detection.

The study has been carried out under guidance of the Operating Room of the Future (FOR) at St. Olavs Hospital in Trondheim, Norway, with the Department

of Circulation and Medical Imaging at NTNU as a close collaborator providing data and expertise in the field of echocardiography.

1.2 Research Goals

The study presented in this thesis is based on two main goals covering implementation and evaluation.

Research goal 1 *Propose and implement a pipeline utilizing deep learning for automatic MAPSE detection on TEE B-mode recordings.*

This research goal is composed of a number of different tasks. Firstly, the main task is designing a pipeline for automatic MAPSE estimation. The power of deep learning and convolutional neural networks has been thoroughly demonstrated in the recent years, and an important aspect of this study has been to utilize deep learning, and investigate its potential for automatic MAPSE detection. The use of deep learning involves tasks such as collecting and annotating data, designing a network architecture, training the network and evaluating its performance. The research goal also stresses the use of TEE B-mode recordings. Since the study was originally proposed by the Operating Room of the Future, using TEE B-mode recordings is motivated by the prevalent use of TEE in the operating room. Finally, the pipeline includes several components solving less complex tasks for deriving a final MAPSE estimate.

Research goal 2 *Compare MAPSE values automatically estimated by the proposed pipeline from goal 1 with manually obtained MAPSE values obtained in a clinically approved environment.*

This research goal essentially consists of two tasks. The first task is to set aside a subset of the available data for testing. MAPSE for every recording in this test set then has to be both manually detected by a specialist in a clinically approved environment, and automatically detected by the proposed pipeline. The second task is the final comparison of the manually and automatically obtained MAPSE values. This will give an indication of how well the proposed pipeline performs. Because the study is blinded, this final comparison will not be performed until the pipeline is complete.

With research goals 1 and 2 as the foundation, this thesis provides an investigation of the feasibility of automatic MAPSE detection using deep learning.

1.3 Prior Work

This section presents relevant prior work related to the research conducted in this study. The main sources of inspiration for this study were methods used for facial landmark detection, deep learning in medical imaging and previous work in automatic assessment of cardiac function.

1.3.1 Deep Learning

In recent years, methods using deep learning have outperformed classical methods in computer vision tasks such as object recognition, segmentation and landmark detection [7]. For this study, landmark detection via regression using deep learning is the most relevant task. The most researched application of landmark detection is facial landmark detection. Duffner et al. [8] were the first to use convolutional learning of Gaussian distributions in a supervised manner to extract five facial landmarks [9]. A method using cascaded convolutional neural networks (CNNs) to refine the localization of five landmarks was proposed by Sun et al. [10], and the method was improved and expanded to detect 68 landmarks by Zhou et al. [11]. Recently, Merget et al. [12] proposed a method for performing heat map regression for facial landmarks using deep learning, outperforming previously mentioned methods by factoring in both the global and the local context of the facial landmarks.

Deep learning methods have also been successfully applied to medical imaging, with segmentation as the most researched application. Ronneberger et al. [13] proposed an encoder-decoder architecture for medical image segmentation called U-Net. By using transposed convolutions and skip-connections between the encoder and the decoder, U-Net, and variations of it, has shown to be an effective architecture for image segmentation overall [14, 15, 16]. Further, deep learning is being heavily researched within diagnostic ultrasound imaging, and more specifically in echocardiography. Dezaki et al. [17] automatically recognized cardiac cycles by using residual convolution layers in combination with recurrent layers. The visual feature extraction from the residual layers combined with the recurrent layers yields a model that can learn both spatial and temporal features. A method combining dynamical models with deep learning for tracking of the left ventricle endocardium for segmentation has been developed by Carniero et al. [18], outperforming the state-of-the-art tracking methods of the time. Sofka et al. [19] proposed a measurement point detector consisting of a fully convolutional network for point localization and recurrent neural layers to refine the point location.

1.3.2 Automatic Assessment of Cardiac Function

Several methods have been used for automatic assessment of cardiac function. The classical methods include speckle tracking and tissue Doppler imaging (TDI). The first method using TDI to assess myocardial velocity was proposed by Isaaz et al [20]. Heimdal et al showed how strain rate could be detected in real-time using TDI [21]. Both TDI and STE are essential tools in echocardiography, although they both have a few limitations. Limitations of TDI as a method for detecting tissue velocity, strain and strain rate detection include angle dependency in the measurements and low signal-to-noise ratio [22, 23]. Speckle tracking echocardiography (STE) was first proposed by Reisner et al. in 2004, for automatic calculation of two-dimensional strain and global longitudinal strain [24, 25, 26]. STE is inherently susceptible to drifting, because it is based on finding corresponding speckle patterns between B-mode frames [27].

Van Stralen et al [28] proposed a method of detecting and tracking the mitral annular plane in 3D echocardiography. Their method involves finding LV long axis center points by applying a Hough transform, and then they project these points through the long axis to a perpendicular plane located at the endocardial border, which is the mitral annular plane. The mean error rate was -1.54 ± 4.31 mm for the mitral annular plane. De Veene et al [29] proposed a different method for tracking the mitral annular plane in 3D echocardiography. By modeling the displacement between subsequent frames, they got a root mean squared error of 1.96 ± 0.46.

Grue et al. [31] automatically detected MAPSE from transthoracic echocardiography (TTE). Using edge detection to fit a model of the left ventricle, they obtained a mean difference of -0.2 mm and a standard deviation of 2.1 mm compared to reference measurements. Using a U-Net architecture, Smistad et al. [30] proposed automatic MAPSE and ejection fraction detection from TTE using deep learning. This was done by performing segmentation of the LV, myocardium and the left atrium. For MAPSE they had a mean difference of -0.9 mm and a standard deviation of 4.6 mm.

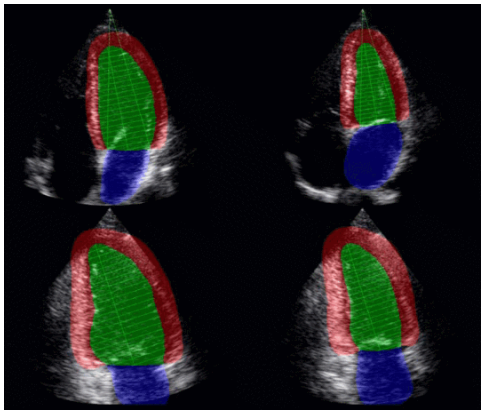


Figure 1.1: Automatic detection of MAPSE using deep learning image segmentation [30].

1.4 Structure of Thesis

This thesis is organized as follows:

- **Chapter 1**
  The introduction chapter provides the background for the study, the research goals and a summary of relevant prior work.

- **Chapter 2**
  Necessary theoretical background for the work presented in this thesis is covered. Cardiac physiology and function assessment is explained briefly. The basics of ultrasound imaging, diagnostic usage and echocardiography are covered. In addition, a detailed overview of artificial neural networks is presented.

- **Chapter 3**
  Methods and materials used to meet the research goals are covered in this chapter. Data acquisition and processing is described. The design and implementation for fully automatic MAPSE detection is presented in detail.

- **Chapter 4**
  The results from the data acquisition, training and testing of the landmark detector, and the final MAPSE results compared to clinically obtained MAPSE values are presented.

- **Chapter 5**
  A discussion of several aspects of the study, from data acquisition and processing, to the specific components of the pipeline, and the overall performance compared to clinical examination is presented. The chapter ends with limitations of the study and suggestions for further work.

- **Chapter 6**
  A summary of the findings presented in the previous chapters concludes the thesis.

2 Theoretical Background

2.1 The Human Heart

The human heart is a muscular organ located within the thoracic cavity. It is responsible for pumping blood through the vessels of the circulatory system. The circulatory system consists of two circuits, the systemic circulation and the pulmonary circulation. The first part of the systemic circulation distributes oxygenated blood throughout the body, supplying the cells of the body with oxygen and nutrients. The second part transports blood low in oxygen back to the heart. The pulmonary circulation transports blood low in oxygen from the heart, through the lungs, where the blood is reoxygenated before it is transported back to the heart.

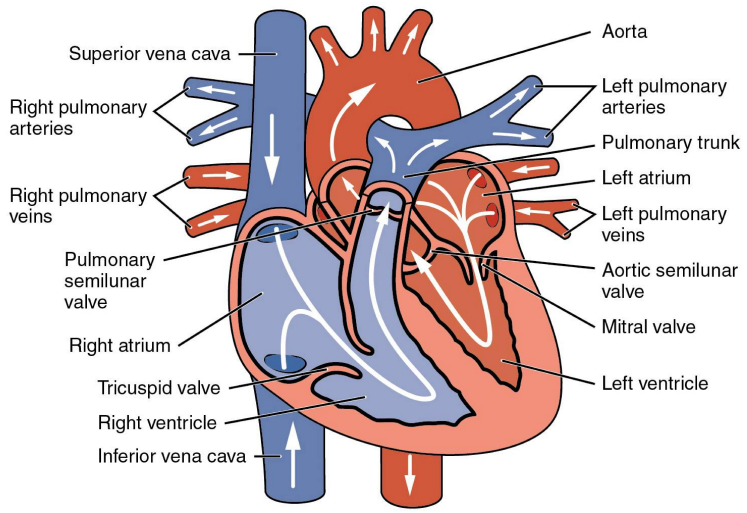


Figure 2.1: The human heart. Figure from [32].

2.1.1 Cardiac Physiology

The human heart, shown in figure 2.1, consists of four chambers; two upper atria and two lower ventricles. The left atrium and ventricle are separated from the right atrium and ventricle by the septa. The blood flow between the left atrium and ventricle is facilitated by the mitral valve, while the blood flow between the right atrium and ventricle is facilitated by the tricuspid valve. The primary function of the valves is to act as one-way valves, preventing backflow while giving as little resistance as possible for forward flow.

The main function of the heart is to maintain pressure in the blood vessels supplying the cells of the body with oxygen and nutrients. By acting as a parallel double pump, the heart pumps out blood and maintains the necessary pressure. Figure 2.2 shows the pressure in the left ventricle and atrium, and the aortic pressure through the heart cycle.

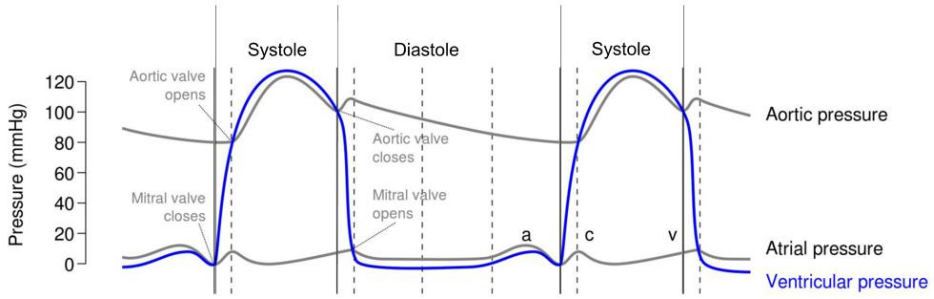


Figure 2.2: The aortic, atrial and ventricular pressure though the cardiac cycle. Modified figure from [33].

In the first phase, the atrioventricular valves are closed, and the heart muscle (myocardium) contracts. The blood flows from the left ventricle through the aortic valve into the systemic circulation, and the blood from the right ventricle flows through the pulmonary valve into the pulmonary circulation. This phase is called the *systole*, and is referred to as the ventricular ejection phase. From figure 2.2 the ventricular pressure is seen to increase rapidly, and the blood subsequently flows through the aortic valve. At end-systole (ES) the aortic and pulmonary valves are closed and the ventricular volume is at minimum. In the second phase of the heart cycle, the left and right ventricles receive blood from the left and right atria respectively, through the atrioventricular valves. This phase is called the *diastole*, and is referred to as the ventricular filling phase. From figure 2.2 it can be seen that the diastole starts when the aortic valve closes and the pressure drops as the

ventricle is filled with blood. At end-diastole (ED), when the mitral and tricuspid valves close, the volume of blood in the ventricles is at maximum.

2.1.2 Assessment of Cardiac Function

Assessment of cardiac function provides important information about the myocardial function, and is done when cardiac disease is suspected, for routine control and monitoring of patients with known cardiac disease, for intensive care patients, during high-risk surgery and during catheter based cardiac interventions.

Perioperative cardiac monitoring in cardiac surgeries typically involve electrocardiography (ECG) and transesophageal echocardiography (TEE) [3]. With ECG, the electrical activity of the heart is determined and irregular ECG patterns are detected, showing signs of cardiac abnormalities.

Echocardiography is commonly used to monitor and assess the left ventricular systolic and diastolic function. To facilitate assessment, the left ventricle is divided into segments shown in figure 2.3.

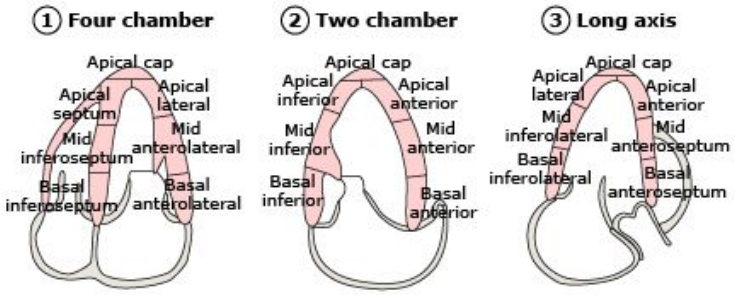


Figure 2.3: Left ventricular segments in four chamber, two chamber and long axis view. Modified figure from [34].

Assessment of LV function can be broadly divided into two main categories; global and local function assessment. Local function assessment provides information of the regional cardiac function and typically measures contraction patterns across the myocardium through the cardiac cycle. Myocardinal strain and strain rate are examples of local function parameters with information of local myocardial deformation of basal, mid and apical segments. Lack of functionality in certain areas can, in turn, indicate that there is a problem in the coronary artery supplying the specific area or presence of disease affecting regional myocardial function.

Global LV function assessment is concerned with the overall function. A widely used measurement of the LV's ability to supply the body with blood, it called *left ventricular ejection fraction* (LVEF). LVEF is a global functional measure of the

percentage of blood leaving the ventricle in the systole, and is calculated by the volumetric difference of blood in the ventricle at ES and ED.

Another global functional parameter is *mitral annular plane systolic excursion* (MAPSE). MAPSE is defined by the atrioventricular plane displacement (AVPD) between ES and ED, illustrated by figure 2.4. It is typically measured in millimeters or centimeters. MAPSE is a commonly used parameter for global longitudinal function assessment and have shown to correlate well with LVEF [35]. MAPSE has also shown to be a sensitive measure for detection of abnormalities caused by early stage cardiovascular diseases where longitudinal function is affected first and the patient may have a normal LVEF [36]. In addition, recent studies indicate that MAPSE is a valuable measure during surgery, where current methods for calculating LVEF are time-consuming [37].

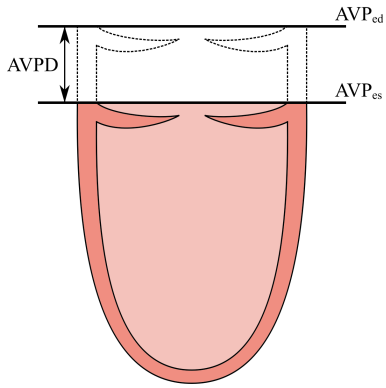


Figure 2.4: Atrioventricular plane displacement between ES and ED.

2.2 Diagnostic Ultrasound Imaging

Over the last 70 years, ultrasound imaging has been continuously developed for use in medical diagnostics. Modern ultrasound technology is now able to accurately measure blood flow in vessels, movement of tissue, and show detailed anatomical structures in 3D [38].

2.2.1 Basic Principles

Ultrasound refers to sound waves with frequencies higher than the human hearing range. The sound waves used in diagnostic ultrasound imaging are longitudinal waves, with frequencies typically in the range of 2 to 15MHz [38]. The waves are produced when electrical current is applied to piezoelectric crystals located in the transducer (probe). The piezoelectric crystals are used for both emission of sound waves and receiving the reflected sound waves as the current producing the vibration is applied in a pulsed manner. When the ultrasound wave propagates through a medium, such as tissue, bone or blood, the particles in the medium

oscillate with the frequency of the wave along the direction of the propagation. At the intersection of two mediums the waves are partly reflected due to the acoustic impedance difference in the mediums, and partly propagated further [39].

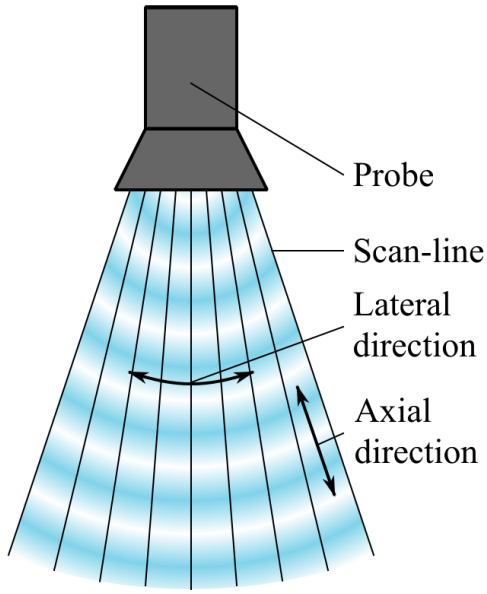


Figure 2.5: Ultrasound probe.

At the off-time of the applied current in the piezocrystals, the reflected waves, referred to as the *echo*, create a vibration in the crystals. This produces electrical current. The strength of the echo determines the amount of current produced. The time from emission until the echo is received determines the distance from the probe. The current produced by the echos across the surface of the probe is amplified and processed, and a grayscale image is generated. The image displays the strength of the echo as pixel intensity. The surface of the probe is divided into scan-lines, from which the horizontal axis of the image is constructed. The vertical axis of the image displays the distance from the probe.

The quality of ultrasound images is determined by both spatial and temporal resolution. Spatial resolution has two main components; the axial and the lateral directions (figure 2.5). The axial direction is parallel to the ultrasound beam, while the lateral direction is perpendicular to the ultrasound beam. Both axial and lateral resolution increase by increasing the frequency of the sound waves [5]. However, due to faster attenuation of sound waves with shorter wavelength and the inverse relationship of frequency and wavelength (equation 2.1), the depth of penetration decreases as the frequency is increased.

$$f = \frac{c}{\lambda} \tag{2.1}$$

f is frequency, c is the speed of sound in the medium the sound wave is propagating through, and λ is the wavelength. Increasing the frequency will give higher spatial resolution but shorten the depth of penetration, and vice versa. Temporal resolution is determined by the frame rate of the recording. Increasing the frame rate will increase the temporal resolution.

2.2.2 Diagnostic Usage

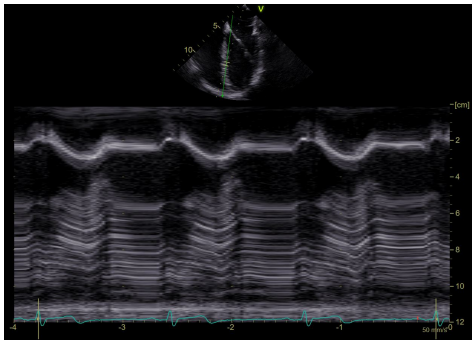
Ultrasound imaging is broadly adopted across many fields within medicine and continuous technological development has made ultrasound an essential diagnostic tool. Compared with other imaging modalities, diagnostic ultrasound imaging has many benefits. It is inexpensive, has an excellent safety record, provides real-time imaging and can measure flow and velocity of blood and tissue [39, 40].

Ultrasound imaging provides different modes, and their usage depends on what needs to be measured or imaged. The three most widely used modes include:

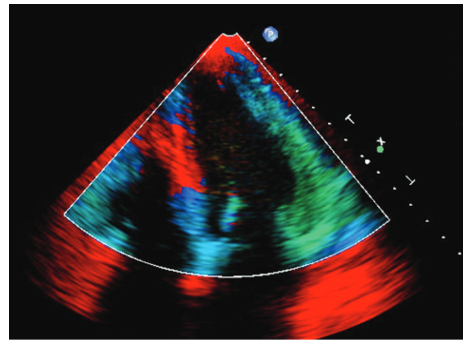
- **B-mode**
  B-mode imaging is short for *brightness mode* and is the most common ultrasound mode. B-mode images are generated as described in section 2.2.1. Figure 2.6a features a B-mode image at the top.

- **M-mode**
  M-mode imaging is short for *motion mode*. M-mode is often used for assessment of tissue movement over time. An M-mode image is obtained by selecting a scan-line in a B-mode image. The intensity from this scan-line is then displayed on the vertical axis and time information for the recording on the horizontal axis. This is shown in figure 2.6a, where the green line in the B-mode image is the scan-line. The corresponding M-mode image is shown in the bottom part of the figure. The current way to derive MAPSE is by detecting the mitral plane at ES and ED in M-mode.

- **Doppler mode**
  Doppler mode imaging can provide information about velocity in bodily fluids and tissue. In order to measure velocity relative to the probe, Doppler mode utilizes the Doppler effect. Tissue Doppler images display the velocities in a color overlay on B-mode images, shown in figure 2.6b, or as tissue Doppler curves.



(a) B-mode (top) and M-mode (bottom).



(b) Color Doppler mode [38].

Figure 2.6: Ultrasound modes.

2.2.3 Transesophageal Echocardiography

Echocardiography has become an essential tool for the assessment of cardiac function. The two most common types of echocardiography are transthoracic echocardiography (TTE) and transesophageal echocardiography (TEE). TTE is the most widely accepted type of echocardiography, performed by placing the ultrasound probe on the exterior of the patient's chest. TEE is performed by inserting the probe into the patient's esophagus. Due to the TEE transducer's close proximity to the heart, the use of higher frequency waves is feasible. This increases the resolution at the expense of penetration depth [42]. In everyday practice, however, the frequencies used for TTE and TEE are in the same range.

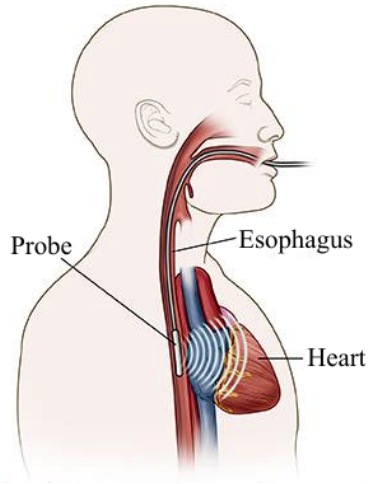


Figure 2.7: Transesophageal echocardiography [41].

As well as being a useful tool to complement TTE, TEE is widely adopted in the operating room. Since the TEE probe can be placed and left in the patient's esophagus, it is currently an essential monitoring tool for cardiac surgeries [43, 3]. By providing high resolution images in real-time, perioperative use of TEE includes monitoring and assessment of cardiac function, and guidance of surgical tools and implants [43].

When performing TEE, the myocardium can be imaged from multiple slices,

commonly called *views*. The views are cross-sections of the heart from different angles. Three common views for TEE recordings are two-chamber (2C), four-chamber (4C) and *apical long axis* (ALAX). Figure 2.8 shows the different cross-sectional angles for the three views. Figure 2.9 shows example images from the three views.

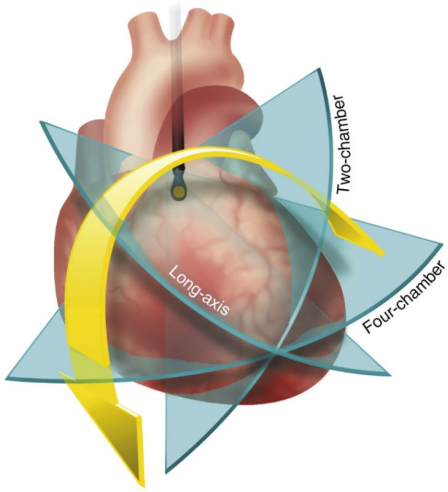


Figure 2.8: Slicing of the heart for three common TEE views [44].

(a) 2C. (b) 4C. (c) ALAX.

Figure 2.9: Three common views for TEE.

Figure 2.9a, shows the complete left ventricle and a part of the left atrium at the top. Figure 2.9b, shows the complete left ventricle, and parts of the left and right atrium, and right ventricle. Figure 2.9c, shows the left atrium and ventricle, as well as the aortic valve and the start of the aorta.

2.3 Artificial Neural Networks and Deep Learning

Machine learning and artificial intelligence have been heavily researched areas of computer science for over 60 years. During the most recent years, variations of artificial neural networks (ANN) has been used to revolutionize certain tasks within the fields of computer vision, speech recognition and natural language processing.

This section will first briefly introduce the field of machine learning. Then, the inner workings of a feed-forward neural network and how to train it will be described. Lastly, convolutional neural networks will be explained in detail.

2.3.1 Basics of Machine Learning

The field of machine learning is concerned with designing algorithms that can learn mappings of the input data to produce a desired output. These mappings are typically too difficult to explicitly program, and the development of learning algorithms has made us able to solve difficult tasks. Satisfactory amounts of high quality data is an essential aspect of successful machine learning systems. There are two main types of ways these systems can learn, and the choice depends on what kind of data is available:

- **Supervised Learning**
  Supervised learning refers to the use of data which has ground truth, or reference, annotations when training the machine learning algorithm.

- **Unsupervised Learning**
  When doing unsupervised learning, the training data does not have a ground truth annotation. Unsupervised learning is often used to find similarities in data, for instance by clustering.

The machine learning method presented in this thesis has been trained by supervised learning, and the rest of this thesis will therefore focus on supervised learning. The two most common tasks solved with supervised learning are:

- **Classification**
  Classification is done by assigning the input data to separate, specific classes based a set of features of the input data. ANNs are very often used for classification tasks. Examples of classification tasks include image classification, image segmentation and speech recognition.

- **Regression**
  Regression is the task of mapping the input data to numerical values based on a set of features of the input samples.

The machine learning algorithm has to be trained to be able to learn a certain mapping for its specific task. Training is done to find a *model* that generalizes to new, unseen data which the system has not been trained on. To measure how well the model generalizes during the training phase, the total available data is typically divided into three datasets [45]:

- **Training set**
  This dataset is used to train the machine learning algorithm. During the training phase, the parameters of the model are adjusted to minimize the error of the output compared with ground truth.
- **Validation set**
  This dataset is used to control the training phase. The parameters of the model are not adjusted to minimize the error of the validation data.
- **Test set**
  This dataset is used after the training phase is completed, used to assess the final performance of the model.

The size of the different datasets varies depending on the amount of available data and what kind of task the model is training for. When the machine learning algorithm minimizes the error on the training data, the validation error will also decrease. However, at one point the validation error will start to increase despite the training error continuing to decrease. This is called *overfitting*, and happens when the model is becoming too specialized on the training data. Figure 2.10 shows a typical training error and validation error plot when training a machine learning algorithm. The optimal point for stopping the training is when the validation error is lowest.

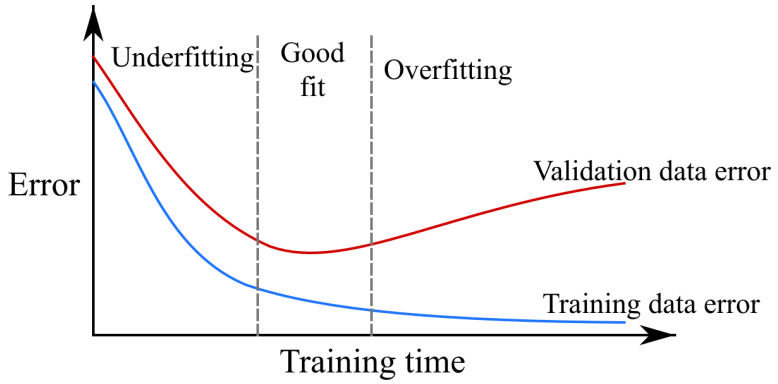


Figure 2.10: Typical training error (blue) and validation error (red).

Evaluation Metrics

A good evaluation metric is essential for evaluating the performance of a machine learning model. The correct evaluation metric to use for a specific task depends on the nature of the task.

The following statistical measures are commonly used for classification and detection purposes:

- **Specificity**
  Specificity, also called *true positive rate* (TPR), measures the how many positive instances are classified as positives. TPR is given by equation 2.2.

$$TPR = \frac{TP}{TP + FN} \tag{2.2}$$

  TP is true positive predictions and FN is false negative predictions.

- **Sensitivity**
  Sensitivity, also called *true negative rate* (TNR), measures the how many negative instances are actually classified as negatives. TNR is given by equation 2.3.

$$TNR = \frac{TN}{TN + FP} \tag{2.3}$$

  TN is true negative predictions and FP is false positive predictions.

- **Matthews Correlation Coefficient (MCC)**
  MCC is a measure that takes into account true and false positives and negatives. The result of MCC is in the range $[-1, 1]$, where 1 corresponds to perfect prediction, 0 corresponds to predictions no better than random guessing, and -1 corresponds to completely imperfect prediction. This measure is balanced, making it very useful if the classes are unbalanced. The MCC measure formula is given by equation 2.4.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.4}$$

2.3.2 Fully Connected Neural Networks

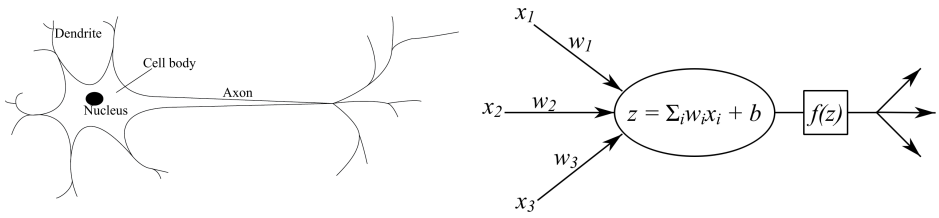
Fully connected neural networks are the simplest type of neural networks and can be used for simple classification and regression tasks.

Artificial Neurons

Artificial neurons are the smallest building blocks of a feed forward, fully connected neural network. The workings of an artificial neuron, as the name suggests, is inspired by a biological neuron. Biological neurons, in general, consist of dendrites, a cell body with a nucleus and an axon. The cell body receives electrical signals through the dendrites. The signals are processed by the cell body and a response is transmitted through the axon. In a similar fashion an artificial neuron receives an input signal, processes it by summing the weighted inputs, and outputs the result. The output is given in equation 2.5

$$z = \sum_i w_i x_i + b \tag{2.5}$$

z is the output, which is given by the weighted sum of the inputs. w is weight, x is input data and b is a correction bias. The bias is included so that the artificial neuron is able to shift the weighted sum [46]. This type of artificial neuron is referred to as a *percetron* and was first developed in the 50's [47]. A graphical representation of the similarity between a biological neuron and an artificial neuron is shown in figure 2.11.



(a) Biological neuron [48].

(b) Artificial neuron.

Figure 2.11: Biological and artificial neuron.

Activation Functions

A network with simple perceptrons is only able to learn linear mappings of the input data [49]. The weighted sum in a neuron must be activated by a non-linear function, an *activation function*, f, for the ANN to be able to learn complex non-linear mappings of the input data. The activated output from a neuron is denoted by a. The most commonly used non-linear activation functions are the following:

- **Sigmoid**
  The sigmoid function is a logistic function which produces an output in the range $[0, 1]$. It is a bounded, differetiable, real function with a non-negative derivative at each point. The sigmoid function is given by:

  $$f_{sigmoid}(z) = \frac{1}{1 + e^{-z}} \tag{2.6}$$

  A problem that can occur in ANNs using the sigmoid function is called the *vanishing gradient problem*. This problem arises due to the shape of the sigmoid function; it is saturated on both sides. The vanishing gradient problem limits the ability of the network to learn from gradient based learning [49].

- **Hyperbolic Tangent**
  The hyperbolic tangent function (tanh) is similar in shape to the sigmoid function, but produces an output in the range $[-1, 1]$. The tanh function is given by:

  $$f_{tanh}(z) = \tanh(z) \tag{2.7}$$

  The tanh function is commonly used in *recurrent neural networks* (RNN). Neural networks using the tanh function also experience the vanishing gradient problem [49].

- **Rectified Linear Unit**
  The rectified linear unit function (ReLU) activates the weighted sum linearly above a certain threshold, usually zero. Below the threshold, the activation is zero. The ReLU function is given by:

  $$f_{ReLU}(z) = max(0, z) \tag{2.8}$$

  The ReLU function is currently the most successful and broadly adopted activation function for neural networks and has proved to make training of neural networks more feasible by avoiding the vanishing gradient problem [50]. The ReLU function is said to be "[...] the single most important factor in improving the performance of a recognition system." [51]. ReLU is also computationally efficient, which speeds up the training process.

Multilayer Networks and Deep Learning

Multilayered ANNs are built by stacking artificial neurons in a layered fashion where neurons of one layer are connected to neurons of other layers. Weights and biases in the ANN are referred to as *parameters*, θ. The first layer of a neural network is called the input layer and the last layer is called the output layer. The layers between the input and output layers are called *hidden layers*. In a fully connected, feed forward neural network, every neuron in a hidden layer receives the output from every neuron in the previous layer, and transmits the output to every neuron of the next layer. Input data is first fed to the input layer, then, passed through the hidden layers and out through the output layer. This is referred to as *forward pass*. The final prediction is the the transformation of the input data after being passed through the network. If the task for the fully connected, feed forward neural network is classification based on a set of features, the number of neurons in the input layer corresponds to the number of features in the input data, and the number of neurons in the output layer corresponds to the number of classes. Figure 2.12 shows a multilayer fully-connected feed forward neural network as a computational graph.

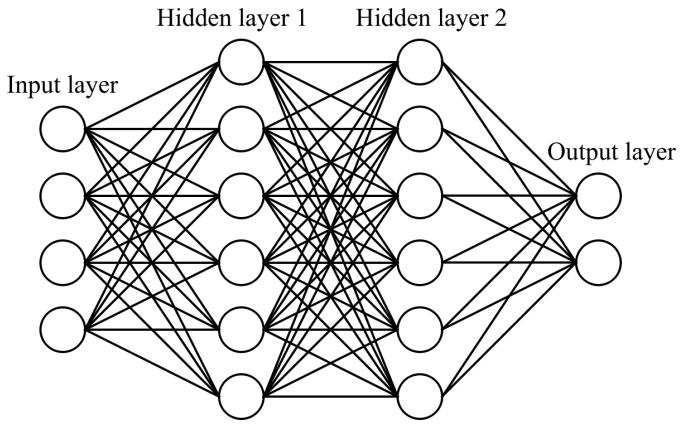


Figure 2.12: Multilayered neural network structure.

The number of layers in an ANN, L, is referred to as the *depth* of the network, and the term *deep learning* refers to training multilayered ANN.

An ANN with at least one hidden layer with non-linear activations, has been shown to be a *universal approximator*, which means that it can be trained to approximate any computable function [52].

There are two other main types of ANN architectures used for deep learning:

- **Recurrent neural network (RNN)**

RNNs are commonly used when the input data contains temporal information, for instance natural language processing or speech recognition.

- **Convolutional neural network (CNN)**
  CNNs are commonly used when the input data contains spatial information, for instance image classification or object detection. CNNs will be explained in detail in section 2.3.4.

2.3.3 Training neural networks

For ANNs to be useful and able to make accurate predictions for their specific task, they need to be trained. During training, the training set is usually divided into a set of *mini batches*. Each mini batch is passed through the network layers as described in the previous section. After a mini batch has been passed through, the parameters of the ANN are updated based on the output error for that mini batch. The size of the mini batches varies depending on total amount of available data, but recommendations say that a mini batch size of 64 samples will give good performance.

There are three necessary components for training neural networks; a *loss function*, the *backpropagation algorithm* and *optimization*.

Loss Function

The *loss function*, or the *criterion*, is the objective the neural network is training to minimize. The basic principle of a loss function is to calculate the error, or the *loss*, between the reference annotation and the predicted output from the ANN. The parameters of the ANN are subsequently adjusted based on how much they contribute to the loss.

Different loss functions are used for different applications and choosing the correct loss function for the specific task is essential for good performance. The following loss functions are the most common loss functions used for training ANNs.

- **Mean Squared Error, L2 Loss**
  Mean squared error (MSE) loss, also called quadratic loss or L2 loss, is given by equation 2.9. This function is used to minimize the mean of the squared distances between the reference and the predicted output.

$$C_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.9}$$

  $\hat{y}_i$ is the predicted output from neuron i in the output layer, corresponding to the activated output from the neuron, a_i^L. y_i is the corresponding reference

annotation.

This loss function is often used in tasks involving regression. Since the difference between the prediction and the reference is squared, predictions far from the reference are penalized more compared with closer values. This can in theory make the last refinement of the prediction error time consuming for deep networks.

MSE, or the root of MSE (RMSE), is also often used as an accuracy metric to assess the performance of a regression model.

- **Mean Absolute Error, L1 Loss**
  Mean absolute error loss, also called absolute loss or L1 loss, is given by equation 2.10. This function is used to minimize the mean of the absolute distances between the target and the predicted output.

$$C_{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{2.10}$$

  L_{MAE} is used for similar cases as L_{MSE}. However, this loss function only calculates the absolute difference between the prediction and the reference, penalizing large and small error gradients equally.

- **Cross Entropy Loss**
  Cross entropy loss is given by equation 2.11. This loss function calculates the error between prediction and reference when they are defined as probability distributions.

$$C_{CELoss} = -\sum_{i=1}^{n} y_i \ln \hat{y}_i + (1 - y_i) \ln (1 - \hat{y}_i) \tag{2.11}$$

  Cross entropy loss is the most common loss function used for classification tasks in deep learning. Typically for classification tasks with multiple classes, the reference vector $\mathbf{y}$ is a one-hot-encoding of the classes, and the output vector $\hat{\mathbf{y}}$ is the predicted probability distribution of the classes [46].

- **Binary Cross Entropy Loss**
  Binary cross entropy loss (BCELoss) is given by equation 2.12. This loss function calculates the error similarly to cross entropy loss, however, it is used when there are only two classes. The output it produces can be seen as a probability of the occurrence of one class.

$$C_{BCELoss} = -\sum_{i=1}^{n} y_i \ln(f_\sigma(\hat{y}_i)) + (1 - y_i) \ln (1 - \sigma(\hat{y}_i)) \tag{2.12}$$

Before the logarithm of the prediction is calculated, the prediction is activated by the sigmoid activation function to squeeze the output prediction in the range $[0, 1]$. BCELoss is often used for image segmentation, but can also be used for regression of probability distributions.

Backpropagation

An essential step in training ANNs, necessary for adjusting the parameters of the ANN to minimize the loss, is to compute the gradient of the loss function with respect to every parameter of the ANN. This is done with an algorithm called *backpropagation*.

Four equations, 2.13-2.16, are essential for computing the gradient of all the parameters of a feed forward, fully connected neural network [46].

$$\delta^L = \nabla_a C \circ f'(z^L) \tag{2.13}$$

$$\delta^l = ((w^{l+1})^\top)\delta^{l+1} \circ f'(z^l) \tag{2.14}$$

$$\frac{\partial C}{\partial b^l_j} = \delta^l_j \tag{2.15}$$

$$\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j \tag{2.16}$$

δ^l is referred to as a neuron error in layer l, C is the loss function, a is the activated neuron output, z is the weighted sum in a neuron, w and b are weight and bias respectively, f is the activation function and $\circ$ is the element wise product of two vectors.

The first equation, 2.13, computes the error of the neurons in the output layer, $l = L$. This is done by finding the element-wise product of the gradient of the loss function with respect to the output from the output-layer neurons, $\nabla_a C$, and the derivative of the activation function in the output-layer neurons, $f'(z^L)$.

The second equation, 2.14, computes the error of the neurons in every layer by applying the chain rule. When the error of the output-layer neurons is computed, the error of the neurons in the last hidden layer can be found by computing the element wise product of the output layer error multiplied with the output layer weights, and the derivative of the activation function of the neurons of the last hidden layer. This way, by applying the chain rule and working backwards though the layers, the error of every neuron can be found. These errors corresponds to the gradient of the loss function with respect to the output of every neuron in the ANN, e.g. the amount that each neuron affects the loss.

The third equation, 2.15, says that the gradient of the loss function with respect to the biases of the ANN is equal to the corresponding neuron error.

The last equation, 2.16, reveals that the gradient of the loss function with respect to one of the weights of the ANN is equal to the error of the neuron of the weight multiplied with the activation from the neuron of the previous layer that the weight is connected with.

By propagating the gradient of the loss backwards through the ANN with the chain rule, the backpropagation algorithm is able to compute how much every parameter of the ANN is contributing to the loss.

Optimization

After the gradient of the loss function with respect to the weights of the network has been found, the weights have to be adjusted to minimize the loss. This is done with optimization techniques. Training of a neural network is often described as searching for a minima of a non convex Θ dimensional hyperplane where Θ is the total number of parameters in the neural network. Figure 2.13 is an illustration of a hyperplane composed of two parameters.

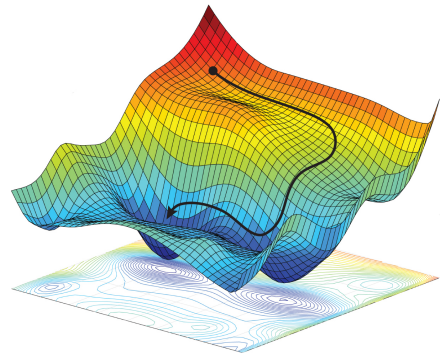


Figure 2.13: The search for the minima [53].

Optimization has been a heavily researched field within machine learning and recent improvements made have had a considerable impact on the training of neural networks. The most common optimization techniques for deep learning are based on variations of gradient descent and are described below.

- **Stochastic Gradient Descent**
  Stochastic gradient descent (SGD) is the most commonly used optimization algorithm in deep learning [49]. When using SGD, a random mini batch of the input data is first sampled and fed through the network. The average loss across the minibatch is then used as an estimate of the total loss. The gradient of the parameters are found from backpropagation and the weights are adjusted according to SGD's update rule, which is given by equation 2.17.

$$\theta \leftarrow \theta - \eta \nabla_\theta C \tag{2.17}$$

  θ is a network parameter, η is the learning rate and $\nabla_\theta C$ is the gradient of the loss function with respect to the parameter θ. The learning rate η determines by which rate the parameter should be adjusted. Increasing the learning rate

often yields faster convergence, however it can result in the ANN being stuck in a local minima. Decreasing the learning rate will on the contrary decrease the rate of convergence, but can make the ANN avoid sharp, sub-optimal, local minima.

- **AdaGrad**
  AdaGrad, short for *adaptive gradient algorithm*, is an adaptive optimization algorithm proposed by Duchi et al [54]. When using AdaGrad, every model parameter is given a learning rate which is individually adapted. The update rule for AdaGrad is given by equation 2.18.

$$\theta_{t+1,i} \leftarrow \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} g_{t,i} \tag{2.18}$$

  $g_{t,i}$ is the gradient of the loss function with respect to every network parameter θ_i at each step t. G_t is a diagonal matrix containing the sum of the past gradients squared. ϵ is a smoothing term included to avoid division by zero.

  Since AdaGrad calculates adaptive learning rates for every network parameter, the need for manually searching for a good learning rate is reduced. However, since G_t increase as the ANN is training, the learning rate can eventually become so small that the network stops learning [49].

- **RMSProp**

  RMSProp, short for *root mean square propagation*, is an unpublished optimization method. The developers of RMSProp address the problem of AdaGrad and propose the following solution; when replacing the diagonal matrix G_t in AdaGrad with a decaying average of all past squared gradients, the learning rate does not decrease at the same rate [49]. The average of all past squared gradients is given by equation 2.19 and the update rule for RMSProp is given by equation 2.20.

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1E[g^2]_t \tag{2.19}$$

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{E_t + \epsilon}} \nabla_\theta L \tag{2.20}$$

- **Adam**
  The Adam optimizer, short for *adaptive moment estimation*, combines the ideas behind AdaGrad and RMSProp. For the Adam optimizer, the decaying average of past gradients are calculated both squared, v_t (equation 2.22), and non-squared, m_t (equation 2.21).

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{2.21}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{2.22}$$

m_t is an estimate for the mean of past gradients, while v_t is an estimate for the uncentered variance. β_1 and β_2 control the decay rate of the moving averages. Since $m_{t=0}$ and $v_{t=0}$ are initialized to zero, the estimates are biased towards zero. To correct for this bias, corrected estimates are proposed in equations 2.23-2.24.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1} \tag{2.23}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2} \tag{2.24}$$

The update rule for Adam is given by equation 2.25.

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \tag{2.25}$$

The Adam optimizer has shown to converge much faster than the previously mentioned optimizers reducing the time needed for training [55]. However, most papers currently published in the field of machine learning use variants of SGD with fine-tuned configurations, which contributes to the fact that a fine-tuned SGD optimizer outperforms every other optimizer.

The Training Phase

The training of an ANN can be summarized by bringing together the loss function, backpropagation and optimization. The following steps are the core steps for training ANNs:

1. Sample a mini batch of the training data and perform forward pass of the mini batch.
2. Calculate the output loss with the loss function.
3. Find the gradient of the loss with respect to every weight and bias of the network from the backpropagation algorithm.
4. Adjust the network parameters by applying the update rule from the optimizer with the corresponding gradient of the loss.

Batch Normalization

Batch normalization is a technique currently adapted and used in every state of the art ANN. By reducing the covariant shift of a hidden neuron, batch normalization improves stability of training ANNs, which greatly reduces the required time spent on training.

After a mini batch of training data has passed through a hidden layer, the output activations from the hidden layer are normalized by subtracting the mean and dividing by the standard deviation. This is referred to as *shifting* the output activations from the layer. However, shifting may result in the adjustments of the weights of the next layer no longer being good. To account for this, batch normalization adds two trainable parameters to every layer which control how much the batch mean and standard deviation affect the output activations.

2.3.4 Convolutional Neural Networks

This section is based on section 2.3.1 from the preliminary project [56].

Convolutional neural networks (CNN) are ANNs with an architecture specialized for processing data with a grid-like structure [49]. This can be one-dimensional time-series data, two-dimensional images or three-dimensional volumes such as point clouds or voxel images [57, 58, 59]. CNNs are composed of convolutional layers, which apply learnable, finite impulse response filters to the input data, essentially performing the mathematical operation *convolution*. When a CNN is trained, the filters in the convolutional layers will learn increasingly abstract spatial features of the input data.

The first successful applications of CNNs were in image processing tasks, where the spatial feature learning in a convolutional layer is inspired by how visual recognition is performed in the mammalian brain [49]. The primary visual cortex (V1) is the first stage of visual processing in the mammalian brain where complex processing is performed. The following three properties of a convolutional layer inspired by the V1:

- **Feature Maps**
  The V1 is structured as a two-dimensional spatial map, where light arriving to a certain location in the retina affects a corresponding area of the V1. Similarly, the input to a convolutional layer is transformed to a feature map where the spatial relationships between the neurons are kept intact, in a two-dimensional structure.

- **Convolution**
  The V1 has *simple cells* with activity that can be described as applying a

linear function on a small area on the input image. In a convolutional layer, this is facilitated by performing convolution on the input image.

- **Pooling**
  The V1 has *complex cells* which are similar to the simple cells. However, they are invariant to small changes in position. This inspires an operation called *pooling* in a convolutional layer.

The idea behind CNNs was first proposed in the late eighties, and the first real-world application with high accuracy was the recognition of hand-written digits [60]. The breakthrough for deep CNNs, however, occurred in 2012 when the deep CNN AlexNet outperformed every other image classification system by halving the error rate on classification of Google image-net [61, 7]. This sparked an enormous interest in the machine learning community and CNNs have been heavily researched the past few years. CNN architectures have proven to perform with outstanding accuracy in tasks such as image classification, object recognition and segmentation. The rest of this chapter will follow the assumption that the input data is two-dimensional image data.

Convolutional Layers

A convolutional layer consists of three main operations: *convolution*, *non-linear activation* and *pooling*.

The first operation is the convolution. In the context of a CNN, it can be described as a filtering of the input data with multiple learnable, finite impulse response filters. The mathematical expression is given by:

$$z_{i,j}^{l} = \sum_{m} \sum_{n} w_{m,n}^{l} a_{i+m,j+n}^{l-1} \tag{2.26}$$

$w_{m,n}^{l}$ is the weight at position $[m, n]$ at layer l where w^{l} is a two-dimensional $k_1 \times k_2$ vector. The weight vector w^{l} is the learnable filter, commonly referred to as a *kernel*. $a_{i+m,j+n}^{l-1}$ is the activated output from layer $l-1$ at position $[i+m, j+n]$. If l is the first layer in the network ($l = 2$), then the activation vector $a^{l-1} = a^{1}$ is the input image and $a_{i,j}^{l-1}$ corresponds to the pixel value at position $[i, j]$.

The activations from the previous layer involved in a convolution, the leftmost gray area in figure 2.14, are referred to as the *local receptive field*. By summing over the product of the kernel and the local receptive field, $z_{i,j}^{l}$ is obtained. By applying the kernel to the whole image in a sliding window fashion, a convolution is performed. The vector z^{l} is often referred to as *feature map* and contains extracted spatial features. Figure 2.14 shows the convolution operation.

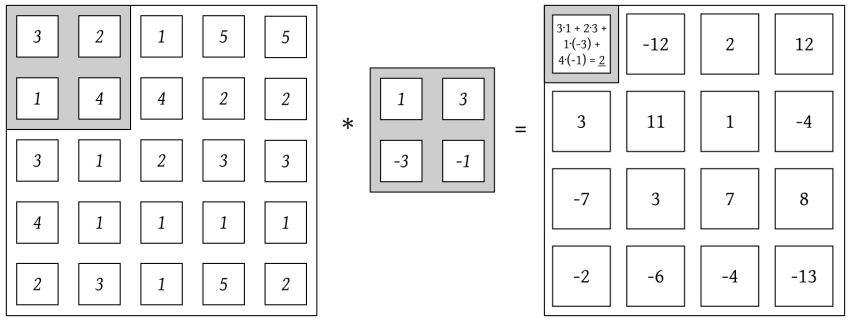


Figure 2.14: Convolution operation.

Non-linear activation is performed on the feature map as described in section 2.3.2. The most commonly used activation function for CNNs is the ReLU function, given by equation 2.8. The general expression for the activation function in a CNN is given by equation 2.27

$$a_{i,j}^{l} = f(z_{i,j}^{l}) \tag{2.27}$$

Figure 2.15 shows the ReLU on a feature map.

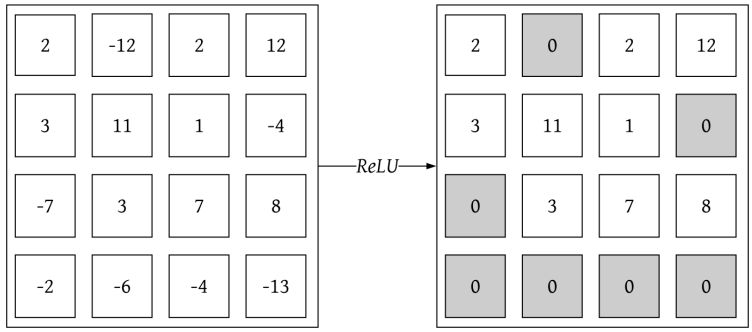


Figure 2.15: ReLU activation.

Pooling of the feature map is performed to reduce the feature representation. The most commonly used pooling operation is called *max pooling*. In max pooling, the highest value in a sliding $m \times m$ window is selected to form a condensed feature map. The output from this operation, a^l, is the output from layer l. Max pooing with a 2×2 grid is shown in figure 2.16.

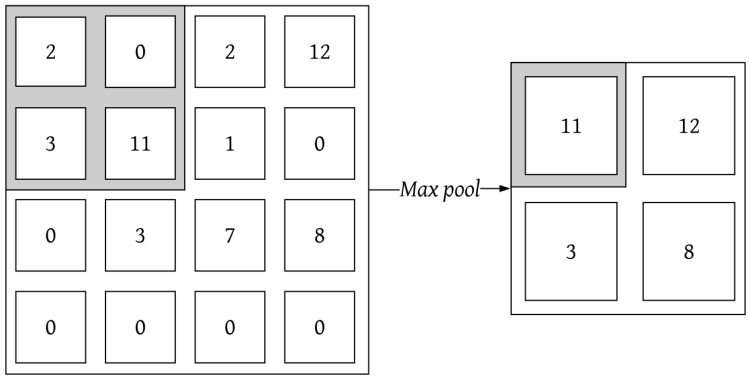


Figure 2.16: Max pooling.

Figures 2.14 - 2.16 show that a typical convolutional layer will perform down-sampling of the input image. The training phase will in turn make the layer able to extract the most important information. When constructing a convolutional layer, one must specify the output size of the layer. The output size corresponds to how many feature maps the layer produces and states how many learnable filters the convolutional layer has.

A different kind of layer used for upsampling of feature maps, is performing an operation called a *transposed convolution*. Figure 2.17 shows how transposed convolutional layers perform upsampling by zero-padding each value of the feature map and applying a learnable filter similarly to standard convolution. The output from transposed convolutional layers are prone to the *checkerboard effect*, where grid-like artifacts are clearly visible in the output [62]. This can be mitigated with smoothing filters.

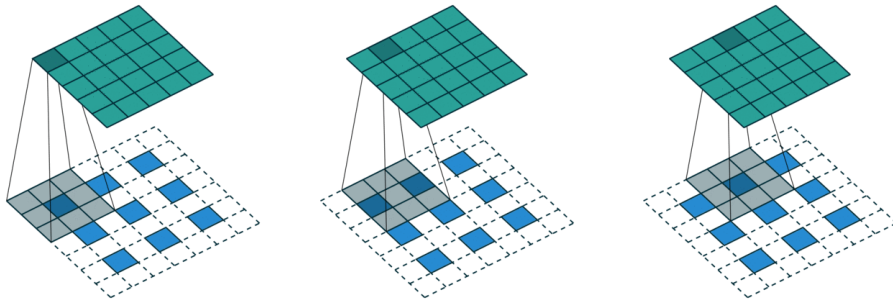


Figure 2.17: Transposed convolution [63].

In recent years, several improvement configurations have been made to convolutional layers and the interaction between them. The introduction of residual layers provided a performance boost for CNNs. Residual layers make use of shortcut connections between convolutional layers, illustrated in figure 2.18. By using these shortcut connections, the CNN layers learn residual mappings of the input data. This enables stacking more layers consecutively, increasing the complexity of the CNN while still improving the feature extraction ability of the CNN. ResNet-152 (containing 152 convolutional layers) beat the human performance in image classification of the ImageNet dataset, which is a dataset comprised of more than 14 million labeled images of 1000 different classes [58, 64].

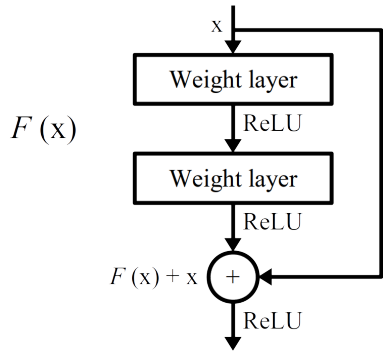


Figure 2.18: ResNet skip connection.

Training a Convolutional Neural Network

CNNs need to be trained to extract spatial features of the input data for the given task.

During the first phase of the training process, the forward pass, image data is passed through multiple convolutional layers as described in the section above. The second phase is where the network learns, which is described in 2.3.3. However, first a few changes have to be made to the backpropagation algorithm for the convolutional layers. The gradient of the loss function with respect to the weights in the kernels of a CNN differs from the gradient of the loss function with respect to the parameters of a feed forward, fully connected ANN. The expression for the gradient of the loss function with respect to every weight in the kernels of the network can be found by applying the chain rule in a similar fashion as described above. In a CNN this is given by equation 2.28

$$\frac{\partial C}{\partial w^l_{i,j}} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \frac{\partial C}{\partial z^l_{i,j}} \frac{\partial z^l_{i,j}}{\partial w^l_{m',n'}} \tag{2.28}$$

C is the loss function. $w^l_{m',n'}$ is the weight at position $[m', n']$ in layer l with size $k_1 \times k_2$. $z^l_{i,j}$ is the pixel value in feature map z at position $[i, j]$ in layer l. The last term, the gradient of $z^l_{i,j}$ w.r.t $w^l_{m',n'}$, can be thought of as the way in which the feature layer z^l is affected when weights in w^l are changed. This is equal to the output from the previous layer, $a^{l-1}_{i+m',j+n'}$. The first term after the summation, the

gradient of the loss function with respect to the feature map z^l, can be thought of as the way in which the loss function is affected when pixel values in z^l are changed. These gradients are referred to as deltas. The gradient of the loss function with respect to the weights can be written as:

$$\frac{\partial C}{\partial w_{i,j}^l} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \delta_{i,j}^l a_{i+m',j+n'}^{l-1} \tag{2.29}$$

Substituting equation 2.16 with equation 2.29 in the backpropagation algorithm makes it useful for training CNNs. Further, the optimizer applies the update rule for the weights as described in section 2.3.3.

Successful CNN Architectures

CNNs are built by stacking convolutional layers sequentially. The following CNN architectures have given significant advancements for different tasks within computer vision which are relevant for this study:

- **VGG**
  The VGG architecture was proposed by Simonyan et al [65] in 2014 and became a very popular feature extractor due to its simple structure. VGG is constructed by 16 layers, 13 convolutional layers and 3 fully connected layers. The convolutional layers use 3×3 kernels. The number of parameters in the network, however, is very large, which means that it consumes a lot of memory when trained.

- **ResNet**
  In 2015, ResNet was proposed by Kaiming He et al [58], and outperformed other architectures in the ImageNet challenge, while still having much less parameters compared with VGG. ResNet feature the skip connections described previously and batch normalization.

- **FCN**
  Fully convolutional network (FCN), proposed by long et al [66], was one of the first successful CNN architecture for image segmentation that only consisted of convolutional layers. FCN consisted of a decoder based on VGG and one upsampling layer performing interpolation.

- **U-Net**
  U-Net, proposed in 2015, is an architecture with multiple transposed convolutional layers in the decoder [13]. U-Net also includes skip connections between the encoder and the decoder. The decoder of U-Net is based on VGG.

3 Method

3.1 Data Acquisition and Preparation

Like research goal 1 states, TEE B-mode recordings of the left ventricle was the data to be used for automatic MAPSE detection. After conferring with Dr. Rye Berg it was decided that the cardiac planes relevant for MAPSE detection were the 2C and 4C views, as MAPSE is usually calculated for 2C and 4C views.

The data set used in this study was recorded at the Echocardiography Unit in the Clinic of Cardiology at St. Olavs University hospital in Trondheim, Norway, by cardiologists with expertise in echocardiography. State-of-the-art clinical scanners, GE Vivid E9, E95 or S70 with a 6VT-D probe (GE Vingmed, Ultrasound, Horten, Norway) were used for recording. The acquired data originated from patients that were referred to the clinic, and all TEE examinations from December 2018 to May 2019 were included in this study. No patient selection has been done.

After acquisition, the recordings were exported to the hospital's image vault. From the image vault, the recordings were anonymized and the raw data was exported to proprietary DICOM files. The raw DICOM files were then sorted and the relevant views were extracted. Thereafter, every image sequence was scan-converted from the original proprietary DICOM format to isotropic 2D images by applying a polar to Cartesian transformation to the ultrasound raw beam data. Finally, the Cartesian image data and corresponding geometric information were exported to HDF5 files. Reading of the final HDF5 data for this study was done with the open source python library *h5py*.

To meet research goal 2, a proportion of the data set was saved for testing. MAPSE was subsequently derived by Dr. Rye Berg in a clinically approved environment with the method described in section 2.2.2. The software used for detection of MAPSE was *EchoPac 202.34* (GE Vingmed, Horten, Norway). The study presented in this thesis has been a completely blinded study, meaning that the reference MAPSE values detected by Dr. Rye Berg were not revealed until the full pipeline was finished.

3.2 Estimation Pipeline

Research goal 1 states that a pipeline is to be implemented for automatic estimation of MAPSE. To get a fully automatic estimation of MAPSE only based on B-mode images, four landmarks on the mitral plane were chosen as the basis for the calculation; two landmarks in 2C view and two landmarks in 4C view. In 2C view, the left and right points were located at the intersection of the mitral plane and the basal anterior and inferior segments respectively. In 4C view, the left and right points were located at the intersection of the mitral plane and the basal lateral and septal segments respectively. Since calculation of MAPSE in a clinical setting is currently performed in M-mode, the following assumption was necessary to continue:

Assumption 1 *The movement of the four chosen mitral landmarks in B-mode imaging corresponds to the tissue movement in one scan-line displayed in M-mode imaging used for MAPSE detection.*

The input to the pipeline is one TEE B-mode sequence, 2C or 4C view, and the output is the MAPSE value for both the left and right sides depending on the view. It was decided that a reasonable abstraction of the pipeline was to divide it into the following two main modules (figure 3.1):

- **Landmark detector**
  This module's objective is to find the location of the mitral landmarks in TEE B-mode images of the heart.
- **Post processing module**
  This module's objective is to estimate the MAPSE value based on the estimated location of the mitral landmarks from the landmark detector.

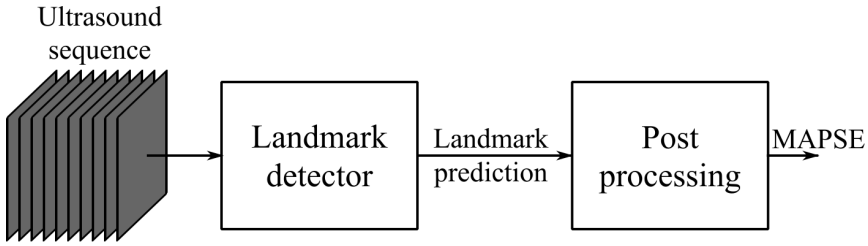


Figure 3.1: Pipeline with landmark detector and post processing module.

The two following sections will describe the method behind the landmark detector and the post processing modules respectively, in detail.

3.3 Landmark Detector

The first module of the pipeline is the landmark detector. It was decided that the method for detecting landmarks was by using a CNN. Two aspects were deemed necessary to take into account when constructing the landmark detector:

1. Predict the location of the two mitral landmarks in 2C (anterior and inferior) and 4C view (lateral and septal).
2. Account for noise in the ultrasound sequence to make the predictions more robust. The model should not predict the location of the mitral landmarks in frames where they cannot be visually located due to noise.

Supervised learning of a fully convolutional neural network was chosen as the method for detecting the landmarks in the TEE B-mode sequences. To account for noise, the landmark detector was going to learn a probability distribution for the localization of the landmarks. This is referred to as *heat map regression* and common practice when performing facial landmark detection [9]. If the recorded sequence was too noisy for the landmarks to be visually detected, all the pixel-values in the reference maps were set to zero. Thus, the landmark detector was expected to learn to estimate the location of the landmarks with a certain confidence. It was also decided that no distinction was made between the left and right landmarks across 2C and 4C views. For the landmark detector, the anterior (2C) and the lateral (4C) landmarks were the same landmark and the inferior (2C) and the septal (4C) landmarks were the same landmark. This was done to reduce complexity and because 2C and 4C views are not significantly different.

3.3.1 Data Preparation for Deep Learning

Before training the landmark detector, certain necessary data preparation steps were performed.

Annotation

Before the landmark detector could be trained with supervised learning and the performance could be assessed, the data had to be annotated. The annotation was performed by the author and Torjus Haukom under supervision from Dr. Rye Berg.

Before the manual annotation was started, it was decided to use B-mode data from the TVI recordings. These recordings provide B-mode recordings which is interleaved with color Doppler imaging. The reason that these recordings were chosen, was due to the low frame rate compared with the frame rate of recordings

with only B-mode imaging, which is about twice as high. By using low frame rate recordings, the amount of frames to manually annotate would be halved.

The annotation was done with a python-script. First, an image sequence was loaded from a h5-file. Then, for every frame in the sequence the annotator would click at the location of the mitral landmarks and a red star would appear where the annotator had clicked. If the landmark was not visible due to noise, the annotator would click in the upper left corner. This landmark would subsequently be annotated as invisible due to noise. To aid higher consistency in the annotation, the location of the landmark in the previous frame was shown with a blue star. To reduce memory consumption for training of the landmark detector, the B-mode sequence and the coordinates of the reference landmarks were saved to new HDF5 files.

Data Set Division

Before training the network the available data had to be divided into a training set, a validation set and a test set. The data for the test set was first randomly selected by patient, in collaboration with Dr. Rye Berg. To get an good indication of the MAPSE estimation of the pipeline, the number of patients in the test set were chosen to be 23. The remaining data was randomly divided, 80% for the training set and 20% for the validation set. An important aspect for data division was to divide by patient and not by single frames due to similarity between frames in one recording of a patient.

3.3.2 CNN Architecture

To be able to visually predict the location of the mitral landmarks while detecting noise, a CNN was chosen for spatial feature extraction. The final architecture was decided to be fully convolutional with an encoder-decoder structure inspired by state-of-the-art architectures used for segmentation and landmark detection [58, 13]. It was decided that the network would predict the landmark for one frame at the time in a single-shot manner.

The encoder part of the CNN was chosen to be the ResNet50 network. This was due to the high performance of ResNet as a feature extractor, the relatively small memory consumption and the availability and simplicity of use with the *PyTorch* framework [58]. The ResNet architecture was modified to be useful for this study. The first convolutional layer was modified to have only one input channel as the input data is grayscale, not colored. The original ResNet50 architecture is built specifically for image classification of the Google ImageNet dataset. Therefore, the output layer is a fully connected layer with 1000 output neurons for the 1000 classes of ImageNet. Additionally, the last operation before the output layer is average

pooling of the feature maps, reducing the feature maps from two-dimensional maps to one-dimensional feature vectors. Since ResNet was being used as an encoder in the landmark detector, the last layer and the last average pooling was removed to keep the spatial relations in the feature maps.

The decoder part of the network consists of transposed convolutional layers with batch normalization and resembles the U-net decoder [13]. The transposed convolutional layers of the decoder were configured to reverse the downsampling of the encoder, reducing the number of feature maps while increasing their spatial size at the same rate as the encoder. This meant halving the number of feature maps and doubling their spatial size for each upsampling block. Unlike similar fully convolutional networks, each upsampling block of the decoder only contained one transposed convolutional layer. This was done due to memory constraints on the *graphical processing unit* (GPU) that the model was trained on.

The output layer from the decoder, was configured to produce two prediction maps with the same size as the input image, one output map per mitral landmark. These maps, along with the feature maps and transformations of the whole network, are shown in figure 3.2.

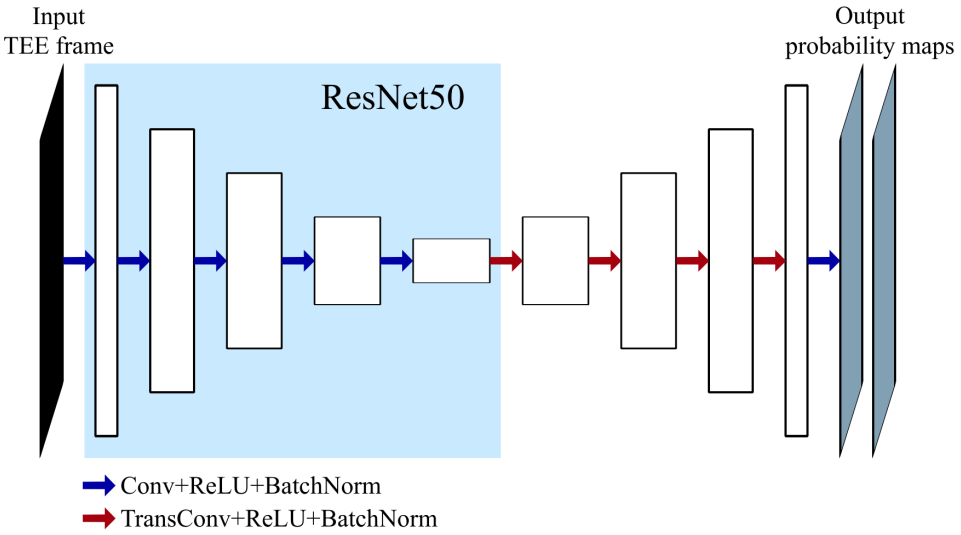


Figure 3.2: Operations and feature maps of the proposed landmark detector.

The future work section from the author's preliminary project suggested to include temporal feature learning to not only detect the landmarks, but also learn the movement of the myocardium to obtain a more robust estimate of the movement of the landmarks [56]. Different approaches were proposed and tested conceptually.

The first proposed method was to try to utilize the temporal feature extraction

of RNNs. The architecture would first consist of a CNN for visual feature extraction. The output of the CNN would then be input for an RNN for temporal feature extraction. The output from the network would be a vector consisting of the predicted landmark coordinates. Preliminary testing of this method did not provide good performing models at all. The model averaged the predicted landmark coordinates for the sequences, essentially performing excessive temporal smoothing. Noise detection was not straightforward for this architecture. The architecture was not feasible for GPU training as the GPU ran out of memory the complexity of the architecture. The complexity stems from a large number of weights in the RNN and the need for several consecutive frames in a B-mode TEE sequence as input for the network.

The second approach was substituting every two dimensional convolution in the encoder with three dimensional convolution; convolution involving x-axis, y-axis and time. This architecture increased the memory size of the encoder in such a fashion that GPU training was unfeasible. This method, however, inspired the configuration experiment described in section 3.3.4.

The network was implemented in *Python* with the *PyTorch* library. PyTorch is an open source machine learning library, currently developed by Facebook. PyTorch's two main features are tensor computation with GPU acceleration and deep learning functionality.

3.3.3 Training the Landmark Detector

Because a CNN was chosen as the method for landmark detection, training was necessary. Preliminary testing showed that the accuracy of the landmark detection did not improve with any significance after 25 epochs. The GPU service that was used for this study had a price per hour used, therefore, the amount of time for training of each configuration had to be limited to not exceed the budget. Therefore, the number of epochs was chosen to be 25. A mini batch size of 32 frames was used as this was the batch size that ResNet50 got the best performance on the ImageNet challenge [67].

Loss Function

The network architecture (section 3.3.2) was configured to output two probability maps, one for each of the two landmarks. Since the cross entropy loss is calculating the error between two probability distributions, it was decided to use this as the loss function for training. However, to avoid having to detect background, binary cross entropy was chosen. By using binary cross entropy separately for the the two output channels, each pixel in the two probability masks were compared to the two reference maps' corresponding pixel. Pytorch's *BCELossWithLogits* method was

used to calculate the binary cross entropy loss of the network output. This method does both sigmoid activation of the prediction and loss calculation for improved stability.

Optimizer and Backpropagation

The optimizer chosen for training the landmark detector was the Adam optimizer. The reason for choosing the Adam optimizer was due to good performance in benchmark testing, fast convergence and simplicity of use. The Adam optimizer is implemented in PyTorch's *optim* module and an instance of the Adam optimizer was created prior to training. Calling the *step* method for the optimizer instance with the loss value as argument initiates adjustment of the parameters of the network.

Backpropagation is implemented in PyTorch by reverse mode automatic differentiation.

Data Augmentation

To fully make use of the available data, it was decided to perform data augmentation when training the landmark detector. For training of the CNN two types of random augmentations were performed in run-time by the PyTorch environment. Each frame was resized so that the shortest side would be 280 pixels before augmentation. The augmentations were:

- **Random Rotation**
  The first type of augmentation was random rotation of the input image. As the angle of the heart's orientation may vary across recordings, augmentation with random rotation was included. When a new image was loaded for the CNN, the image would be randomly rotated by $[-\phi, \phi]$ degrees. $\phi = 10$ was used for training. Random rotation was implemented in python. The image was rotated about the image center with the *rotate* method from the *scikit-image* python library. The coordinates of the reference landmarks were rotated about the image center, $[x_c, y_c]$, with equation 3.1.

$$\begin{bmatrix} x_{rot} \\ y_{rot} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.1}$$

  Figure 3.3 shows a sample image with reference landmarks before and after rotation.

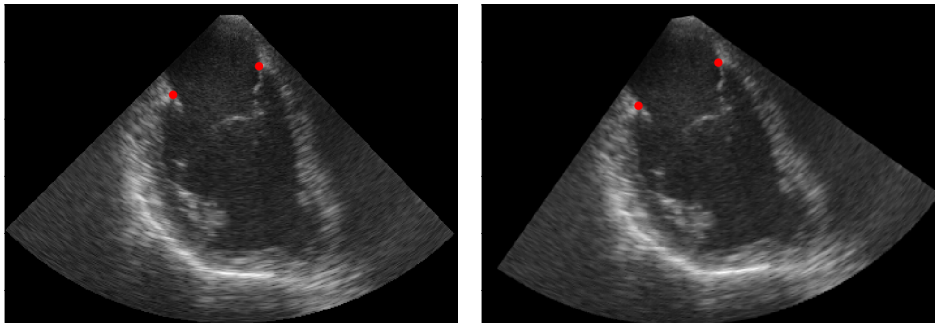


Figure 3.3: Before (left) and after (right) image and reference annotation rotation.

- **Random Crop**
  The second augmentation was random crop. By cropping images randomly the network was assumed to not overfit as early as if the images were identical for every epoch. After an input image was randomly rotated, the image with size 280×280 was randomly cropped to 256×256. The new top and left indices were given by equations 3.2 - 3.3 respectively.

$$i_{top} = randint([0, h_o - h_{new}]) \tag{3.2}$$
$$j_{left} = randint([0, w_o - w_{new}]) \tag{3.3}$$

  After the new top and left indices were found the image was sliced with these indices. Random crop was implemented in Python.

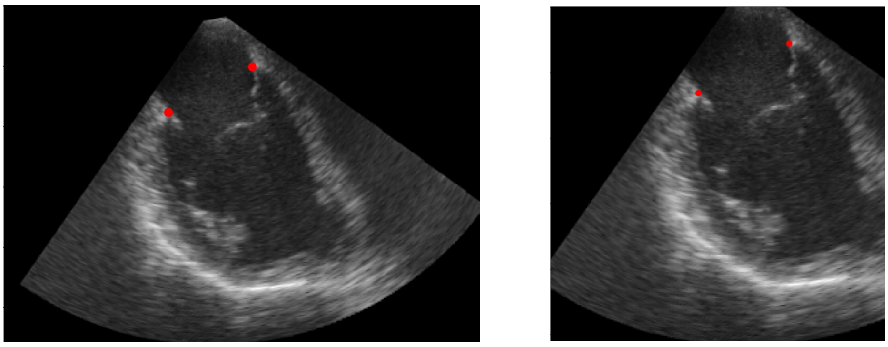


Figure 3.4: Before (left) and after (right) image and reference annotation cropping.

Accuracy Assessment

It was decided to log the mean error distance between the prediction and the reference as well as the noise classification abilities of the network to keep track of how the network is training in terms of accuracy. From the predicted probability maps, the coordinates of the center of mass were found, and the mean error distance was calculated. As for the noise classification, the number of true and false positives and negatives were logged.

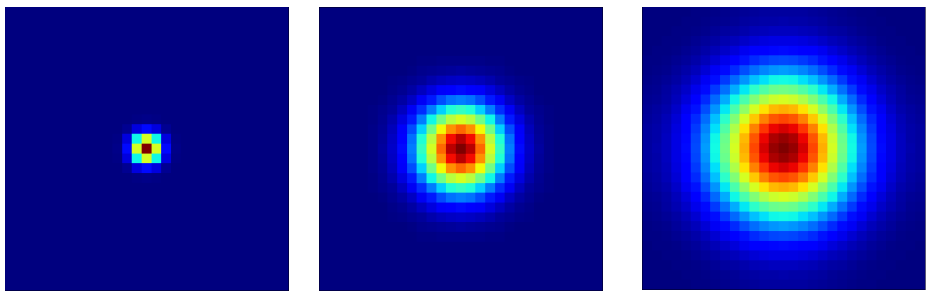
3.3.4 Configuration Experiments and Parameter Search

In order to find an accurate landmark detector with the available resources, it was decided that the network architecture described in section 3.3.2 was to be trained with different configurations and parameters. The configurations and parameters described in this section were chosen based on what were expected to improve the performance of the model for landmark detection.

For evaluation of the different landmark detector configurations, they were assessed on the test set and compared to the reference annotation. Mean absolute error distance for Euclidean, horizontal and vertical distances were used to measure the localization accuracy. TPR, TNR and MCC were used to assess noise detection abilities. MCC was included because the the occurrence and absence of noise in the frames are not balanced. The results are presented in section 4.2.3.

Reference Probability Map Standard Deviation

The first configuration that was trained and tested with different values was the standard deviation for the reference probability maps, σ_{ref}. For a two-dimensional image, the value of σ_{ref} is the standard deviation in pixels. Figure 3.5 shows how varying σ_{ref} influenced the pixel value distribution with a heat-map.



(a) $\sigma_{ref} = 1$. (b) $\sigma_{ref} = 3$. (c) $\sigma_{ref} = 5$.

Figure 3.5: Different σ_{ref}.

Five different values for σ_{ref} were chosen for training and the results were assessed. The standard deviations were $\sigma = [1, 2, 3, 4, 5]$. The standard deviation that gave the highest performing model was chosen as the standard deviation used in the following experiments.

Pretrained Weights

The next experiment was training with and without pretrained weights in the encoder part of the network to assess if pretrained weights provide better performance. The encoder, ResNet50 described in section 2.3.4, was downloaded with pretrained weights from PyTorch's database. The weights have been trained on Google's ImageNet dataset and the model has learned features that makes it able to accurately classify images from 1000 different classes. The network with the pretrained encoder was trained and the results were assessed and compared with the model without pretrained encoder. The performance of the two models decided if pretrained weights were included for the following experiments.

Learning Rate

The third experiment involved varying the learning rate, η for the Adam optimizer. The previous experiments used the pre-defined learning rate recommended by the researchers behind the Adam optimizer, $\eta = 0.001$. Two additional learning rates values, $\eta = 0.01$ and $\eta = 0.0001$, were used for training new models and the results were assessed and compared with the model trained with the baseline value $\eta = 0.001$. The learning rate that provided the highest performing model was chosen as the learning rate to be used for training in the last experiment.

Temporal Convolution Input Layer

The last experiment involved incorporating temporal information for better landmark detection. Different ways to incorporate temporal information were proposed.

The incorporation of temporal information was done by substituting the first layer of the encoder with a layer performing three-dimensional convolution. Similar to the network with a standard encoder, the input image is passed through the network and the location of the landmarks are predicted at the output. When the input layer is performing three-dimensional convolution, the input consists of the image to predict the landmarks for and a number τ of frames before and after that image in the sequence. The output from this layer is two dimensional, and this is achieved by defining a kernel size of $2\tau + 1$ in the temporal dimension. Figure 3.6 shows a sequence of ultrasound images with $\tau = 2$, and how the three-dimensional convolution operation works.

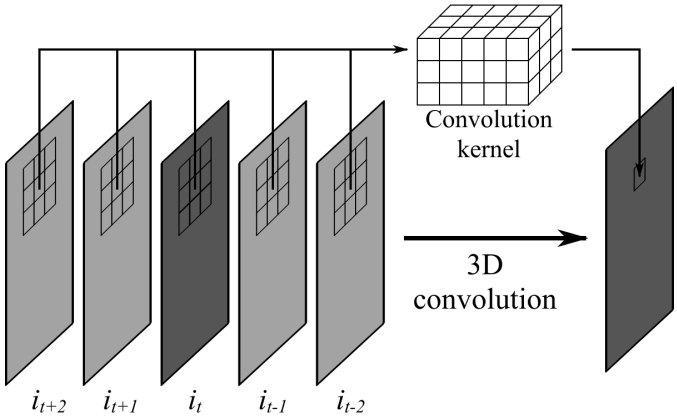


Figure 3.6: Three-dimensional convolution.

The values for the number of frames before and after the current frame to be estimated that were used for the experiment were $\tau = [1, 2, 3]$. The two best performing models were chosen as the models used as landmark detector in the pipeline for automatic detection of MAPSE.

3.4 Post Processing Module

The post processing components are necessary for transforming the probability maps from the landmark detector into MAPSE values. Figure 3.7 shows a block diagram of the components. These components were deemed necessary to obtain the most accurate MAPSE estimates.

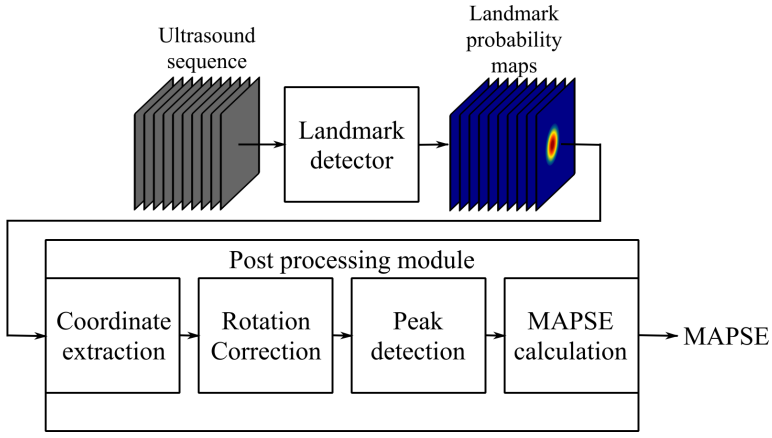


Figure 3.7: Pipeline with post processing components.

3.4.1 Coordinate Extractor

- **Input:** Two probability maps, one for each landmark, for every frame of the TEE B-mode sequence.
- **Output:** The x- and y-coordinates of both landmarks in every frame of the sequence.

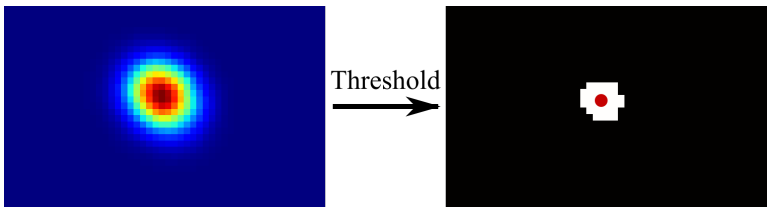


Figure 3.8: Coordinate extraction from thresholding and center of mass calculation.

To obtain the coordinates from the predicted probability maps, the centers of mass of the probability maps after thresholding were calculated. Every pixel with a value above a certain threshold in the prediction mask is set to 1, and the rest is set to 0. The coordinate of the predicted landmark corresponds to the center of mass of every pixel with value equal to 1. The threshold used in the pipeline was 0.5. Figure 3.8 shows thresholding of the probability map and resulting center of mass marked with a red dot.

This component was also used in assessment of the configuration experiment results presented in section 4.2.3.

3.4.2 Rotation Correction

- **Input:** The x- and y-coordinates of both landmarks in every frame of the sequence.
- **Output:** The rotated y-coordinates of both landmarks in every frame of the sequence.

The second component in the post processing module is rotation correction. The TEE images do not always capture the heart in a perfect vertical orientation. Another problem is that the motion of the mitral plane is not always perpendicular to the plane's orientation. The rotation of the two mitral landmarks had to be corrected independently. This was been done by finding k landmarks in ED, where the y-coordinate is at the minimum, and k landmarks in ES, when the y-coordinate is at the maximum. A vector between the mean ED and ES landmarks was spanned

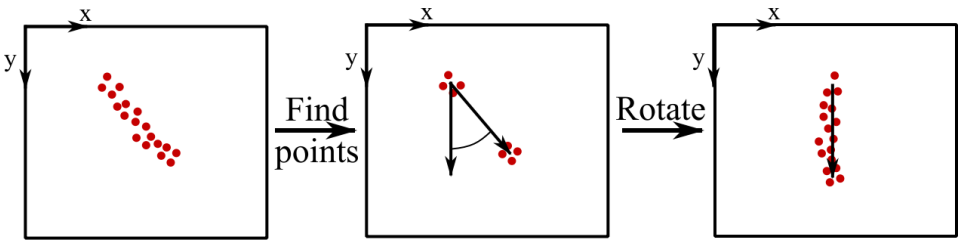


Figure 3.9: Rotation of every landmark in the sequence.

and normalized, and the angle between the vertical axis and this vector was found. The landmarks were subsequently rotated by this angle, illustrated by figure 3.9. Since MAPSE is a calculated displacement between ED and ES, no translation correction was necessary. The output from the rotation correction component was only the y-coordinates of the two landmarks.

3.4.3 Peak Detection and Filtering

- **Input:** The rotated y-coordinates of both landmarks in every frame of the sequence.
- **Output:** Detected ES and ED peaks for both landmarks.

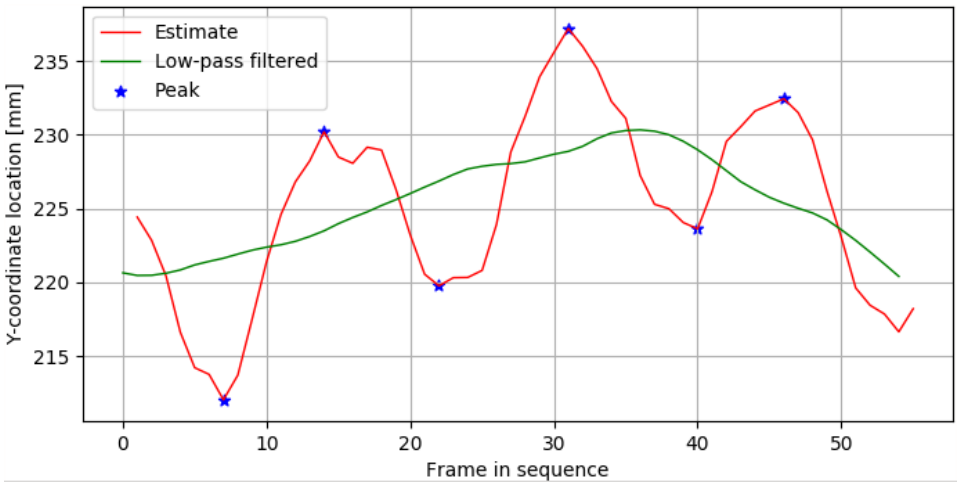


Figure 3.10: Estimated movement of a landmark, the low pass filtered movement and the peaks.

To obtain the final MAPSE values, the distance between end systole and end

diastole had to be calculated. This was done by finding the maximum and minimum coordinate values in the TEE sequence. These peaks also had to be filtered. The peaks were found with SciPy's peak detection method from the signal processing package. To account for local extremes, a minimum distance of 6 frames was used between peaks. When the patient breathes during examination, the whole heart moves a few millimeters. To account for these breathing variations the heart sequence was low-pass filtered. The maximum peaks had to be above this low-pass filtered sequence, and the minimum had to be below. The last step in the peak detection and filtering component was to obtain an equal number of minima as maxima, and have them alternating. This was done with a simple filtering algorithm which carefully selected the correct peaks for the final MAPSE calculation.

3.4.4 Final MAPSE Calculation

- **Input:** Detected ES and ED peaks for both landmarks.
- **Output:** MAPSE for both landmarks

The final MAPSE value for one mitral landmark is given by equation 3.4, which calculates the distance between the mean ES and ED peaks locations, corrected for pixel to millimeter difference.

$$MAPSE = \alpha \left[\frac{1}{P_{ES}} \sum_{i=0}^{P_{ES}} p_{ES,i} - \frac{1}{P_{ED}} \sum_{j=0}^{P_{ED}} p_{ED,j} \right] \tag{3.4}$$

α is the pixel to millimeter correction coefficient, P_{ES} and P_{ED} are respectively the number of maxima and minima, p_{ES} and p_{ED} are respectively the maximum and minimum peaks.

4 Results

4.1 Data Acquisition and Preparation

Table 4.1 shows how the acquired data was divided into datasets for training, validation and test.

Data set	Recordings #	Frames		
		#	2C [%]	4C [%]
Train	131	8520	49.82	50.18
Validation	32	1782	49.89	50.11
Test	46	2879	50.16	49.84
Total	209	13181	49.91	50.09

Table 4.1: Recordings, frames and percentage of 2C and 4C frames in the training, validation and testing sets.

Table 4.1 reveals that a total number of 209 TEE B-mode recordings are included in this study. This amounts to a total of around 100 patients. All the different data sets are seen to contain a very similar number of 2C and 4C frames.

Neither patient selection nor visual inspection of the recordings were performed for any of the datasets. Figure 4.1 shows a patient sample from 2C and 4C view.

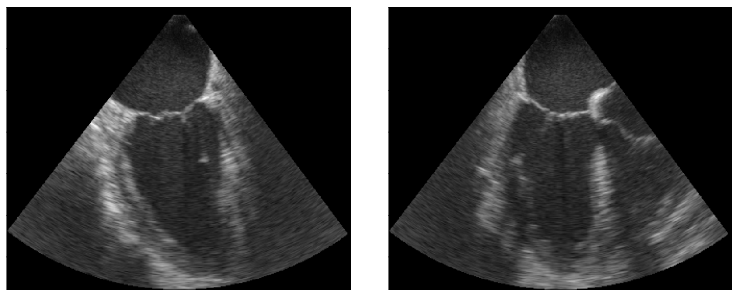


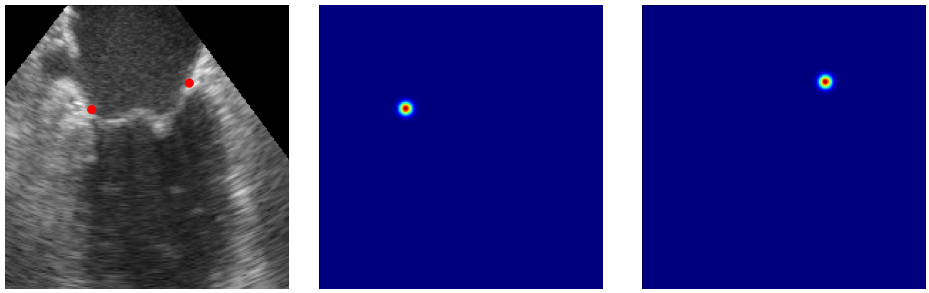
Figure 4.1: TEE images from 2C view (left) and 4C view (right).

Every TEE B-mode frame used in this project has been mirrored. This was not intentional, but happened due to switched axes in the h5py files from the preparation process. When the files were annotated, every image was transposed before they were saved, which caused the images to be mirrored.

4.2 Landmark Detector - Deep Learning Results

4.2.1 Data Annotation

A total number of 13181 single frames were manually annotated. Figure 4.2a shows the reference coordinates after using the python script for annotation. Figure 4.2b-4.2c shows the corresponding probability masks.



(a) Annotated 2C image. (b) Left reference heat map. (c) Right reference heat map.

Figure 4.2: Resulting reference coordinates and probability maps.

Noise artifacts made the landmarks in some frames impossible to visually locate. They were subsequently annotated invisible due to noise, and table 4.2 shows the percentage of the different landmarks that were annotated as invisible.

Data set	2-chamber		4-chamber	
	Anterior [%]	Inferior [%]	Lateral [%]	Septal [%]
Train	6.35	4.75	10.01	3.16
Validation	1.13	7.31	9.00	2.10
Test	0.76	3.81	14.22	1.18

Table 4.2: Percentage of landmarks that annotated as invisible due to noise.

4.2.2 Training the Landmark Detector

The following figures show the logged data from training the model with configurations $\sigma_{ref} = 3$, $\eta = 0.001$, $\tau = 0$ and not pretrained weights in the encoder. Figure 4.3 shows the loss on the training set and figure 4.4 shows the mean error distance for the validation set and the training set.

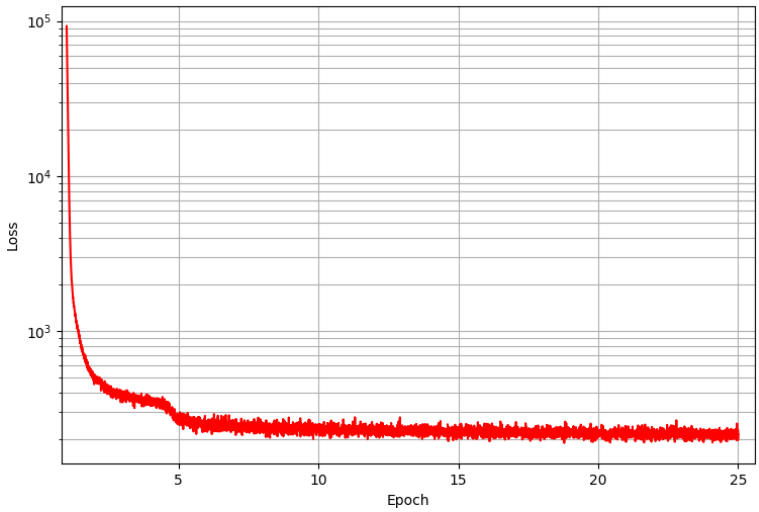


Figure 4.3: The loss of the training set.

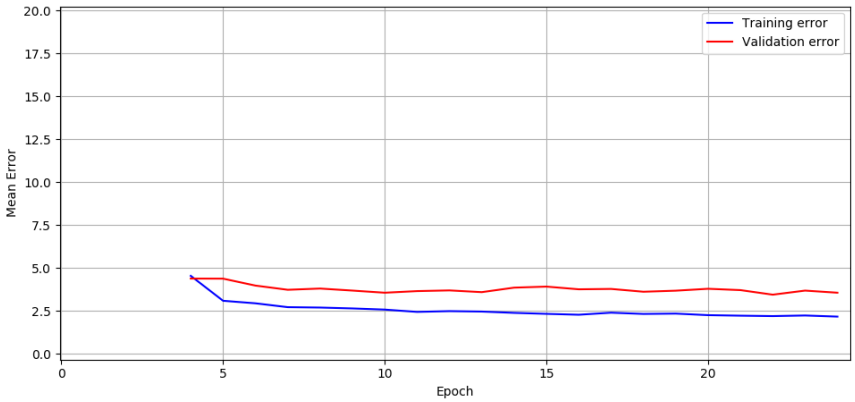


Figure 4.4: The mean error of the training and validation sets.

The reason the error values in the plot in figure 4.4 start in epoch four is because the landmark detector is classifying every landmark as noise in the first three epochs.

4.2.3 Configuration and Parameter Search

The following results show the mean error distance in pixels (MAE) and the noise detection on the test set of the configurations described in section 3.3.4. The distances are mean Euclidean, x-axis and y-axis distance, between the reference annotation and the model predictions. The best results are marked in bold.

Reference distribution standard deviation

The error distance and noise detection of the different standard deviation for the probability maps σ_{ref} are given in table 4.3.

Configuration	Distance [*pixels*]			Noise		
	d_e	d_x	d_y	TPR	TNR	MCC
$\sigma_{ref} = 1$	–	–	–	**1.00**	0.00	0.00
$\sigma_{ref} = 2$	5.48	4.75	1.90	0.85	0.86	0.42
$\sigma_{ref} = 3$	**5.28**	**4.64**	**1.75**	0.82	0.95	0.58
$\sigma_{ref} = 4$	5.48	4.73	1.90	0.71	0.97	0.61
$\sigma_{ref} = 5$	5.48	4.76	1.88	0.65	**0.98**	**0.68**

Table 4.3: Error distance and noise detection for different σ_{ref}.

Reference probability maps with $\sigma = 1$ was not able to find any landmarks with a certainty above 0.5 during the 25 epochs it was trained. Every landmark in every frame of the test set was therefore classified as not detectable due to noise. The model with $\sigma_{ref} = 3$ has the lowest error distance from reference point. The model with the highest standard deviation, $\sigma_{ref} = 5$, had the highest TNR and MCC. The following tests use reference probability distribution maps with $\sigma_{ref} = 3$.

Pre-trained Weights

The error distance and noise detection ability of models with an encoder with and without pretrained weights are given in table 4.4.

Configuration	Distance [*pixels*]			Noise		
	d_e	d_x	d_y	TPR	TNR	MCC
Not pretrained	5.28	4.64	**1.75**	**0.82**	0.95	**0.58**
Pretrained	**5.22**	**4.51**	1.80	0.66	**0.96**	0.54

Table 4.4: Error distance and noise detection for models with and without pretrained weights.

The model with pretrained weights has a lower error distance than the model without pretrained weights, except for error on the y-axis. The model without pretrained weights, however, has the best noise detection ability. The following tests use an encoder without pretrained weights.

Learning Rate

The error distance and noise detection ability when training with different learning rates are given in table 4.5.

Configuration	Distance [$pixels$]			Noise		
	d_e	d_x	d_y	TPR	TNR	MCC
$\eta = 0.01$	5.40	4.79	**1.72**	**0.86**	0.91	0.50
$\eta = 0.001$	**5.28**	**4.64**	1.75	0.82	0.95	**0.58**
$\eta = 0.0001$	5.44	4.76	1.82	0.76	**0.96**	0.57

Table 4.5: Distance error and noise detection for different η.

The model trained with learning rate $\eta = 0.001$ has the shortest Euclidean and x-axis error distance, as well as the highest MCC. The following training used a learning rate $\eta = 0.001$.

Temporal layer

The error distance and noise detection ability for models where the input layer is performing a temporal convolution is given in table 4.6.

Configuration	Distance [$pixels$]			Noise		
	d_e	d_x	d_y	TPR	TNR	MCC
$\tau = 0$	5.28	4.64	1.75	**0.82**	0.95	0.58
$\tau = 1$	**5.27**	4.64	**1.71**	0.66	0.96	0.54
$\tau = 2$	5.56	4.76	2.00	0.69	0.95	0.52
$\tau = 3$	5.33	4.58	1.88	0.74	**0.97**	**0.60**

Table 4.6: Distance error and noise detection for different τ.

The model with input layer performing three dimensional convolution with $\tau = 1$ has the shortest error distance. However, it does not perform noise detection as well as the model without temporal input convolution and the model with $\tau = 3$. The two models chosen for the pipeline for MAPSE estimation is the model with $\tau = 0$ and the model with $\tau = 1$

4.3 MAPSE Estimation

4.3.1 Final MAPSE Results

Landmark Detector Model Comparison

Table 4.7 presents the final MAPSE results from the pipeline with two different landmark detectors, and MAPSE results calculated from the reference annotations, compared to clinically obtained MAPSE. Only the recordings where MAPSE was detected is included in this table and table 4.8 shows the number of excluded recordings. *LD* is short for *landmark detector* in the following tables.

Configuration	MAPSE [mm]			
	2C		4C	
	Inferior	Anterior	Septal	Lateral
$LD_{\tau=0}$	0.45 ± 1.80	0.77 ± 1.65	-0.12 ± 1.50	-0.11 ± 2.18
$LD_{\tau=1}$	-0.08 ± 1.24	0.18 ± 1.31	-0.24 ± 1.81	-0.18 ± 1.50
Reference	-0.21 ± 1.71	-0.45 ± 1.71	0.45 ± 1.21	0.61 ± 2.41

Table 4.7: Mean MAPSE error and standard deviation for every landmark.

The model with $\tau = 1$ has the best performance with an average error of $-0.08 \pm$ 1.38 mm. The model with $\tau = 0$ has an average MAPSE error of 0.25 ± 1.78 mm. When using the reference annotation as the coordinates for MAPSE calculation, the average error is 0.1 ± 1.76 mm. From table 4.7 it can be noted that there does not seem to be systematic error across the models and the reference annotation.

Configuration	2C		4C	
	Inferior	Anterior	Septal	Lateral
$LD_{\tau=0}$	1	1	3	3
$LD_{\tau=1}$	1	1	1	1
Reference	1	1	4	4

Table 4.8: Recordings where MAPSE could not be detected.

Table 4.8 shows the number of recordings of the test set where MAPSE was not detected. The landmark detector model with $\tau = 1$ was able to detect MAPSE in every recording except two, one 2C and one 4C. When using the reference coordinates, MAPSE cannot be detected in a total of 5 recordings, which account for almost 11% of the test set. For reference, MAPSE was detected in every test set recording by Dr. Rye Berg.

Error Distributions

Figures 4.5 and 4.6 shows the prediction errors for the MAPSE values of the test set when using the two landmark detector models with $\tau = 0$ and $\tau = 1$ respectively, with mean and standard deviation.

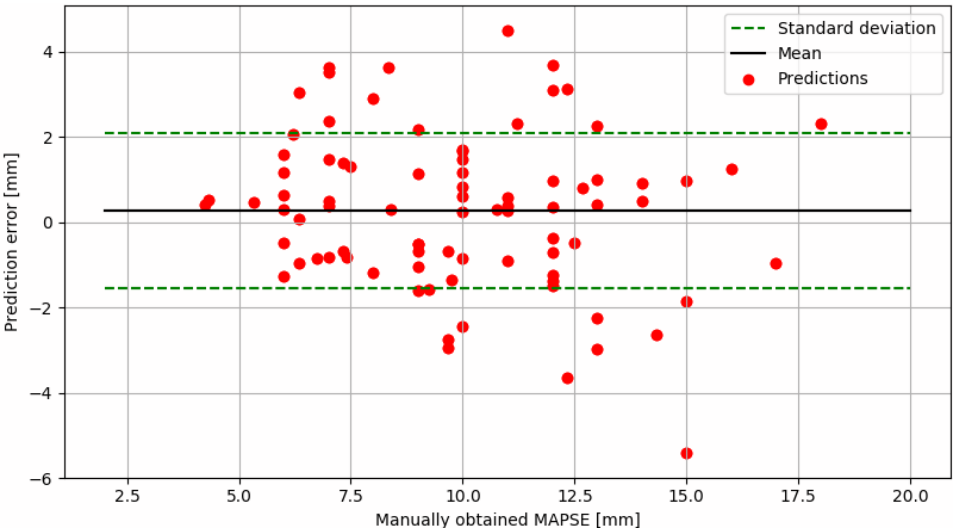


Figure 4.5: Prediction error on the test set when using $LD_{\tau=0}$.

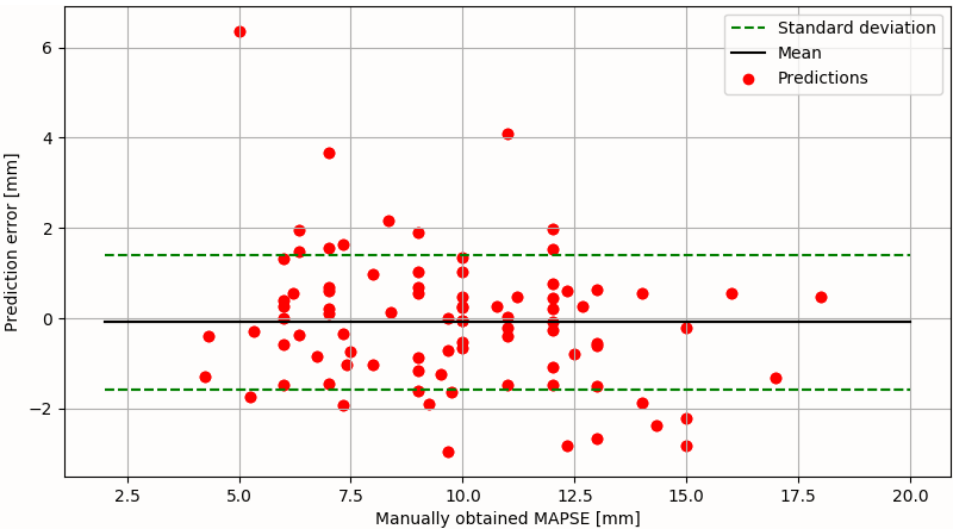


Figure 4.6: Prediction error of the test set when using $LD_{\tau=1}$.

Figure 4.7 shows the cumulative error distribution when using the two landmark detector models. The horizontal axis shows the absolute error distance and the vertical axis shows the proportions of MAPSE detections. The figure shows that the area under curve is much greater for $LD_{\tau=1}$.

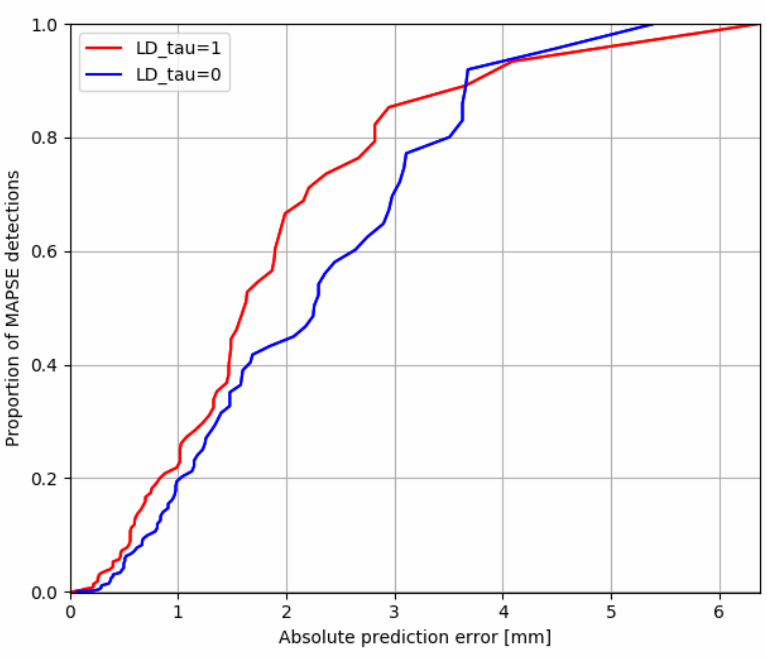


Figure 4.7: Cumulative error distributions when using the two landmark detector models.

4.3.2 Sample Cases

Two different sample cases from the test set are presented in this section, one case with the highest MAPSE estimation error compared with clinical results, and one case where MAPSE was accurately detected. For the following cases, $LD_{\tau=1}$ was used as landmark detector due to better performance.

Both these cases, as well as two other cases, are attached as video files showing the tracking ability of the landmark detector for both noisy and normal scenarios.

Not Accurate Estimate

Figure 4.5 shows one clear outlier with prediction error of 6.37 mm, or 127%. This MAPSE value was calculated from a septal landmark. Figure 4.8 shows ED and ES images from the recorded 4C sequence.

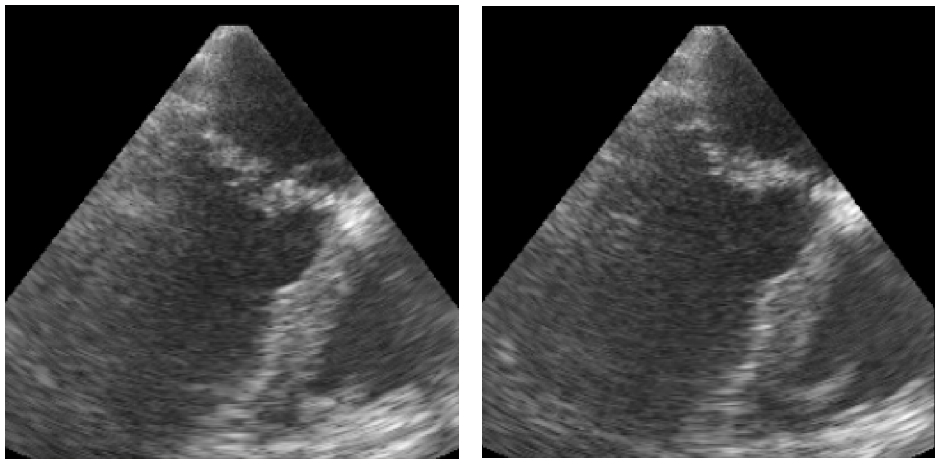


Figure 4.8: ES (left) and ED (right) frames.

Figure 4.9 shows the movement of the landmark when detected and the peaks that the MAPSE calculation was based on.

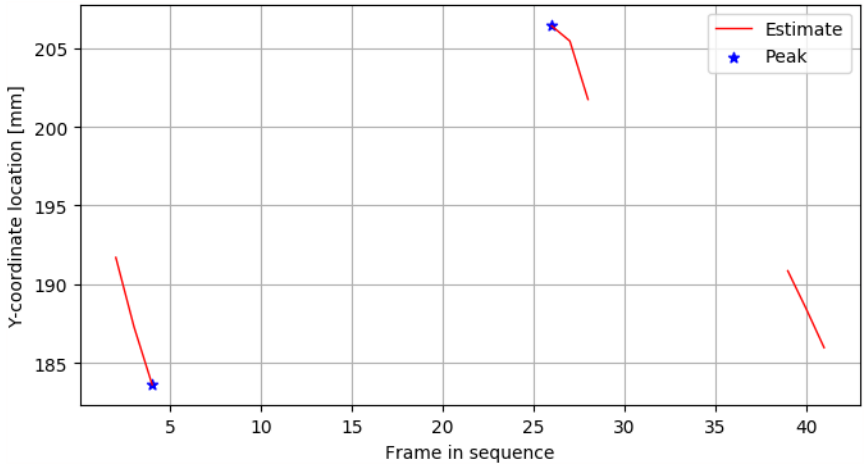


Figure 4.9: Landmark movement and peaks for a non-accurate MAPSE estimate.

This recording, with predicted landmarks, is named *bad_ mapse.avi*, in the attached zip file.

Accurate Estimate

As figure 4.5 shows, many of the MAPSE prediction have a small error. Figure 4.10 shows ES and ED frames of a sequence where the MAPSE estimation error was zero.

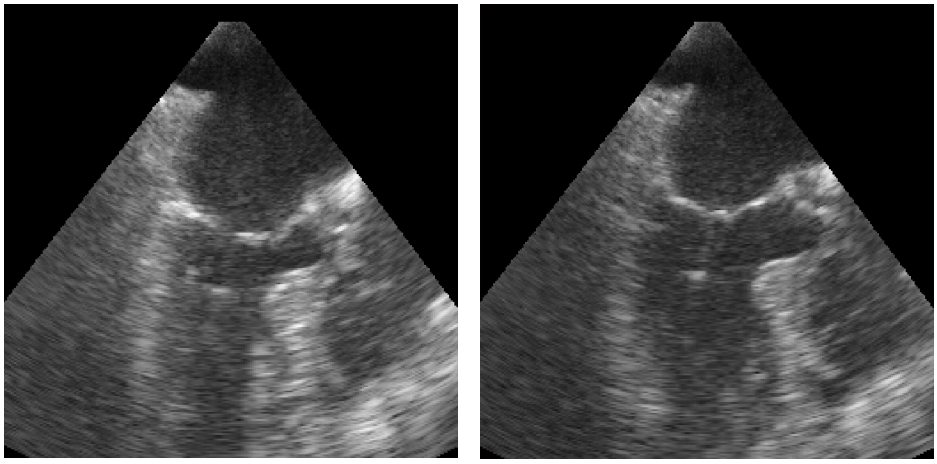


Figure 4.10: ES (left) and ED (right) frames from a high quality recording.

Figure 4.11 shows the movement of the landmark and the peaks that the MAPSE calculation was based on. Significant drift can be seen. This is likely due to breathing.

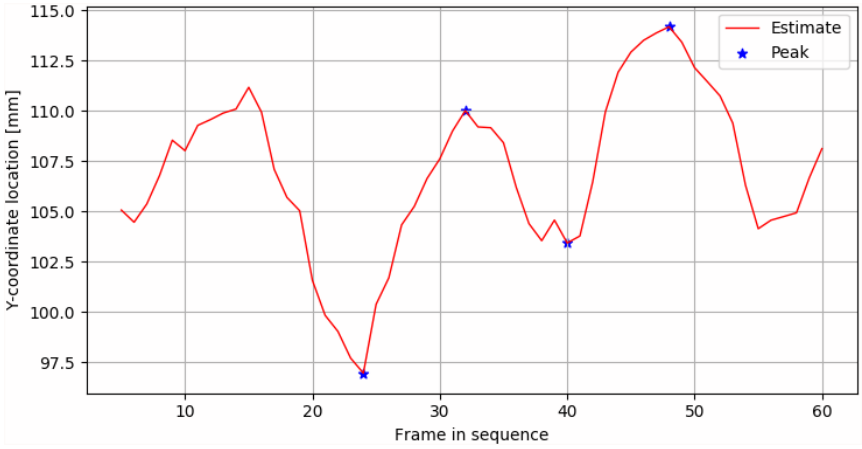


Figure 4.11: Landmark movement and peaks for an accurate MAPSE estimate.

This recording, with predicted landmarks, is named *good_ mapse.avi*, in the attached zip file.

5 Discussion

5.1 Data Acquisition and Pre-processing

From the data acquisition and preparation processes described in section 3.1 it can easily be concluded that the process has been very resource heavy.

For this study, data selection has not been performed; all obtained and available TEE recordings has been used. The recordings in this study were obtained from patients with a wide range of cardiac diseases undergoing a TEE as part of standard care. This means that the patient data included in this study contains recordings with, for instance, heart arrhythmia such as atrial fibrillation. The data used in this study therefore represents data that the system would be exposed to in a real-world setting.

Since there is no existing study trying to automatically detect MAPSE in TEE B-mode recordings (as far as the author is aware of), there does not exist a data set for comparing the pipeline from this study. Hopefully, the data obtained in this study will be used for further research.

5.2 Landmark Detector

5.2.1 Data Annotation

As mentioned section 3.3.1, the annotation of the reference location of the mitral landmarks were performed by two master students under supervision from Dr. Rye Berg. None of the master's students have had previous experience with echocardiography and assessment of cardiac function. Since B-mode images from TVI recordings were chosen as the data in this study, the decreased frame rate may affect the performance of the landmark detector and the pipeline negatively. This is partly because using the higher frame rate would give more training data, and partly because lower frame rate may not capture the full movement of the left ventricle.

Table 4.2 shows that the lateral landmarks in 4C view had the highest proportion of invisible landmarks with 10.74% of the total landmarks annotated as noise. In 2C view the anterior landmark overall was the hardest to manually annotate in the training data, according to table 4.2. For the test data, only the lateral landmark has a high proportion of invisible landmarks. The distribution of noise across the landmarks and the different data sets are important factors for the performance of the landmark detector. In addition, since the distribution of noise across the training, validation and test sets are varying in great measure (as seen in table 4.2), this suggests that the size of the datasets may not be representative for the true distributions of noise. If datasets in deep learning are biased or unbalanced, there is no way of avoiding a biased or unbalanced model.

5.2.2 CNN Architecture

The CNN architecture was chosen based on state-of-the-art architectures used for segmentation, image classification and landmark detection [13, 58, 9].

The feature extractor used as the encoder in this study was the ResNet50. PyTorch offers several different networks that have beaten ResNet in the ImageNet challenge, such as ResNet versions with more convolutional layers, versions of DenseNet, and InceptionV3 [58, 68, 69]. Because these networks are highly optimized for the task of image classification, the potential improvement for this project may not be very significant.

The decoder was inspired by U-Net, but without skip connections between the encoder and the decoder. The importance of both short and long skip connections for medical image segmentation has been demonstrated by Drozdzal et al [13]. They show that including both short and long skip connections gives the best performance. Ronneberger et al, who proposed the U-Net architecture, argues that the long skip connections will make the network able to include higher resolution features from higher resolution parts of the encoder when constructing the output [13]. The landmark detector architecture proposed in section 3.3.2 includes short skip connections by using ResNet as the encoder, however, long skip connections have not been included. This was mostly due to a trade-off between model complexity and mini batch size. Including the skip connections in the architecture may make the model able to refine the predicted locations for the predicted landmarks.

Several methods were proposed for incorporation of temporal feature extraction. However, most of the suggested methods were unfeasible to train with the resources available. Yet, by substituting the input layer with a three dimensional convolutional layer, an attempt for temporal feature extraction was tested. The results does, however, not show significant improvement (as seen in table 4.6). By only having one layer, the first layer in the network, incorporating temporal learn-

ing over only very few time steps, the increase in performance cannot be expected to be significant.

Although including temporal feature extraction may make the model better able to recognize the movement of the myocardium and produce more robust detections, there are certain benefits with one-shot detectors producing detections only based on individual images. Temporal smoothing as a possible artifact of using temporal feature extraction will result in underestimation of MAPSE, and the negative impact of temporal smoothing for left ventricular function assessment has been shown [70]. Another benefit of not having incorporated temporal information in the landmark detector architecture used in this study is that the low temporal resolution data used for training does not affect the performance. The landmark detector should be able to perform just as well with high temporal resolution data.

5.2.3 CNN Training

Training of the landmark detector required a lot of GPU memory. This resulted in having to prioritize between the training batch size and the model complexity. A batch size of 32 has shown to give ResNet50 best performance on the ImageNet challenge, and the batch size for training of the landmark detector was therefore chosen to be 32 [67].

Figure 4.4 shows that the mean error is not recorded before the fourth epoch. After inspecting the output from the network in the first few epochs, it was found that landmark detector classified every landmark to be noise. This likely happened because the fastest way for the network to minimize the loss in the start of the training phase is by setting every pixel in the output probability maps to zero. Since the two prediction maps that the network outputs only has 37 pixels with a value above 0.5, while 99% of the pixels equals zero, predicting every pixel to be zero is a fast and easy way to minimize the loss. After a few epochs, the network starts to predict both landmarks in both of the maps, and at the same time the loss decreases rapidly.

The number of epochs, 25, for training every configuration was chosen from preliminary testing and resource limitations. The mean error on the training set is seen to continue to decrease, while the validation error remains relatively stable. The model is not clearly overfitting on the training data because the validation error does not start to increase during the 25 epochs. The model could possibly have reached a lower validation error by increasing the number of epochs.

5.2.4 Configuration and Parameter Search

Table 4.3 shows that the model trained with $\sigma_{ref} = 3$ achieves the lowest error for Euclidean, horizontal and vertical distance. This may be due the probability distributions being large enough for the network to be able to converge, yet small enough that the landmark location became ambiguous. This claim can be substantiated by the fact that the model with $\sigma_{ref} = 1$ was not able to find any of the landmarks in the test set, predicting only noise. Another takeaway from table 4.3 is that as σ_{ref} increases, the true positive rate of noise detections decreases while the true negative rate increases. This is likely due to the amount of pixels in the reference probability maps being non-zero. When the σ_{ref} was increased, the model learnt a larger probability distribution. One way of dealing with this is to create normalized distributions, where the sum of the probability distribution is constant with varying σ_{ref}.

Table 4.4 shows that the network with pretrained weights achieves a lower error for Euclidean distance due to lower horizontal distance. As the vertical distance is the most important measure for MAPSE, pretrained weights were not used for the following experiments. The model with pretrained weights did not outperform the model without, likely due to the substantial difference between the task that the weights were trained on solving and the current study task. When the difference between tasks is significant, pretrained weights may often halt the performance and in some cases perform worse due to the phenomenon *negative transfer* [71].

Table 4.5 shows that the model trained with $\eta = 0.01$ has a marginally smaller error in the vertical distance compared with the other models. However, the Euclidean and the horizontal distances are not as small as the model trained with $\eta = 0.001$. Since the Adam optimizer implements an adaptive learning rate for every network parameter, the η may have less value compared with other optimizer like standard SGD.

Table 4.6 shows that changing τ does not give systematic difference with respect to the results. The models with $\tau = 0$ and $\tau = 1$ are nearly identical in terms of mean error, however the model with $\tau = 0$ performs better at noise detection. For the pipeline, these two models were compared.

Hardware resources have been limited, and an exhaustive configuration search with cross validation and averaging the results over multiple training runs has not been feasible to conduct. However, the results from section 4.2.3 provide an indication of how changing certain hyperparameters and configurations may affect the performance of the model.

5.3 Post Processing Module

5.3.1 Components

Thresholding and center of mass calculations were used for coordinate extraction. This method seems to be working as intended. Applying a threshold will essentially clip the highest values of the probability maps, as every pixel value above 0.5 will be equally weighted. A potential problem occurs when the location of a landmark is ambiguous and the corresponding probability map has more than one probability distribution. If the result after thresholding is two separate blobs, the center of mass will subsequently be somewhere between the these blobs depending on their size. Setting the landmark coordinates equal to the coordinates of the pixel with the highest value in the probability maps, the mean, is a way of solving this issue. However, if the probability maps contain multiple distributions, the mean with the highest peak may not reflect the true coordinate of the landmark.

Rotation correction was included to calculate MAPSE based on the most correct movement of the landmarks. If rotation correction was not included, the MAPSE values would be underestimated, which was expected.

The peak detection was necessary for finding the displacement of the landmarks between ED and ES. The peaks were also heavily filtered. By setting a minimum distance between peaks to 6 frames, local extremes were avoided. The period of one cycle was commonly somewhere between 15 and 20 frames. However, as the peak detection did not handle NaN values, e.g. frames where a landmark was not found, only the frames with detectable landmarks were used as input to the peak detection method. By having a minimum distance of 6 frames, the local extremes were avoided. The landmark position sequence was also low pass filtered, and the filtered sequence determined the minimum height for the ES peaks and the maximum height for the ED peaks. This also prevented ambiguous local extremes. Because MAPSE is calculated by the displacement between ED and ES, the filtering of the peaks that provided an equal number of alternating ED and ES peaks was necessary to emulate how MAPSE is detected manually. Due to the patient's breathing during the examination, the position of the heart is likely to shift slightly. A potential scenario could be that only the first ED peak was detected, while all three ES peaks were detected. As the patient was breathing, the position of the heart would change, as would the ES peaks in image coordinates. Calculating MAPSE with ES position averaged over 3 peaks and ED position with only one peak, with breathing effects might produce a poor estimate of MAPSE. The last filtering will prevent this by removing the two last ES peaks.

The final MAPSE calculation receives the location of the landmarks at ES and ED, with equal amounts of ES and ED peaks, corrected for myocardium rotation.

5.4 MAPSE Estimation

The results from section 4.3.1 show that the pipeline with landmark detector model with $\tau = 1$ is able to predict MAPSE with a mean error of -0.08 ± 1.38 mm on the test set, compared with clinical assessment of MAPSE. The pipeline does not give significant systematic error for clinical purposes. This grants the importance of every component in the pipeline, and confirms that the pipeline is working as intended.

An interesting result (table 4.7) is found where the MAPSE estimation based on the landmark coordinates of the reference annotation is less accurate compared with the best performing model. The reason for this may be that the landmark detector has learned a more consistent representation of the positions of the different landmarks, whereas the variability of the manual annotation performed by the two annotators was higher. The landmark detector would then learn to locate the landmarks by the generalization of the manual annotations. Another explanation may be that the amount of test data is not sufficient for drawing these conclusions.

Table 4.8 shows that the pipeline with $LD_{\tau=1}$ is not able to estimate MAPSE for only one 2C and one 4C sequence, while the pipeline with $LD_{\tau=0}$ and reference annotation is not able to detect MAPSE in 4 and 5 sequences in total, respectively. This is likely because $LD_{\tau=1}$ has a lower TPR for noise detection, which can be seen in section 4.6. As MAPSE could be detected clinically for all recordings in the test set, the reference annotation for noise detection may not be accurate. It was, however, expected that a landmark detector model with better noise detection ability would be more confident in their prediction and produce results with a smaller standard deviation. Table 4.7 shows that this is not the case for these models. This may be due to the stochasticity of training the landmark detector models. When only training the different configurations once, the performance of the resulting models may not reflect the true relationships between, in this case, noise detection ability and confident landmark predictions.

There does not seem to be systematic tendencies in the mean errors across the difference views in table 4.7. One tendency is that the error of inferior prediction is lower than the anterior prediction for every configuration. Table 4.8 shows that 4C recordings more often are not feasible for MAPSE detection for the pipeline. This can likely be explained by table 4.2, which reveals that over 14% of the lateral landmarks were manually annotated as noise, meaning that the 4C recordings in the test set is likely to contain more noise compared with the 2C recordings.

The results from this study shows that the proposed pipeline performs better than the two other studies, presented in section 1.3.2, doing automatic MAPSE detection [31, 30]. However, this pipeline is trained and tested for TEE, while the previous methods were trained and tested for TTE. TEE may provide images

where the mitral annulus is more clearly detectable, and the task of automatic MAPSE detection may therefore be more feasible when using TEE data compared with TTE data.

The first case presented in section 4.3.2 shows how the worst MAPSE estimate in the test set was calculated and the recording it was calculated from. When inspecting the recording with the output probability map for the septal landmark, it can be seen that the landmark detector confuses the mid part of the mitral valve with the basal septal segment in systole. As seen from figure 4.8, the mitral valve is above the basal septal segment, resulting in a substantial overestimation of MAPSE for the septal landmark. Figure 4.9 shows that the landmark was detected solely in 9 out of 41 frames. For recordings like these, the pipeline should not provide a MAPSE estimate.

The second case in section 4.3.2 shows a recording where the septal landmark was accurately detected and the MAPSE error was zero. Figure 4.10 shows that the image quality at ES and ED is good, and that the mitral plane is easily detectable. For recordings like these, the MAPSE error should be low.

5.5 Limitations

The work presented in this thesis is subject to certain limitations. The following limitations are the most significant.

The first limitation is the amount of available data and the resources needed to obtain more relevant data. Table 4.1 shows that a total number of 209 TEE recordings have been acquired for the purpose of this study. These recordings stem from around 100 patients. To get a satisfactory model using deep learning, data is the single most important resource. In order to acquire data for this study, echocardiographers at St. Olavs have produced more sequences than necessary for the assessment of their patients as TTE is used more often for cardiac function assessment. The cost of acquiring data is a great limitation for the performance of the deep learning based model used in this study. This is also not factoring in the cost of manually annotating of every frame of every recording, as this was part of the study. Additional data preparation, described in section 3.1, has also required the time and efforts of two researchers at NTNU.

The second limitation is the extremely high paced development and improvement of new methods and algorithms within the machine learning community. The work presented in this thesis aims at including the best performing methods for similar tasks, while also trying to specialize the configurations for this study. However, with the current development rate, new methods constantly appear.

Another important limitation, closely related to the deep learning methods,

is the limited computational power available. Several of the deep learning models proposed for this study require computational power above what has been available for use at NTNU. Due to limited hardware resources, a fully exhaustive search for the optimal deep learning model has not been possible to conduct.

5.6 Future Work

The results presented in chapter 4 show that the proposed pipeline detects MAPSE values which correlate to MAPSE values obtained in a clinically approved environment. This study has been a conceptual study investigating the feasibility of automatic MAPSE detection from TEE recordings and there are several important areas for further study.

One of the most important focus areas for future work will be related to data. Since the method proposed in this study is based on deep learning, the amount of training data is the most important factor for the performance of the pipeline. Recently, methods for *active learning* have been proposed, where new data is annotated by an already trained model and incorporated into the training set for training a new model. This can lighten the burden of manual annotation, but requires that the trained model performing the annotation can provide information of how accurate the annotations are. When the active learning model encounters a recording where it fails to correctly annotate the landmarks, it will need to notify the users to manually annotate the specific recording. The landmark detector proposed in this thesis is performing landmark detection while also performing noise detection. This can be used to provide information of the quality of the recording and facilitate the annotation accuracy measure required for active learning.

Another important topic for further work is to continue testing different architectures for the landmark detector. New research and methods constantly appear, providing the possibility of finding a compromise between complexity and accuracy. The proposed pipeline has not been optimized for speed nor memory consumption.

Since this study was a blinded study, improvements is likely to be found by fine-tuning the post processing components to produce more accurate MAPSE values. Incorporating the information of presence of noise will also be an important focus point. By, for instance, setting a minimum required number of frames with detectable landmarks, outliers like the one in section 4.3.2 could be avoided.

The current pipeline only uses B-mode data. The performance of the pipeline may be improved by incorporating different types of data. An example is including ECG data to accurately detect when ES and ED occur, and only detect landmarks in these frames. By detecting landmarks in only ES and ED frames, the time required for automatic MAPSE detection will be reduced.

6 Conclusion

The basis of the work presented in this thesis are the two research goals 1 and 2. The first research goal stated that the aim of the project was to propose and implement a pipeline for automatic detection of MAPSE using deep learning. The second research goal stated that the proposed pipeline needed to be evaluated against clinically obtained MAPSE values.

A pipeline for automatic detection of MAPSE was proposed and implemented with a deep convolutional neural network as a landmark detector. The landmark detector was trained to find the location of two mitral landmarks in 2C and 4C TEE B-mode recordings. The pipeline also included necessary post processing components to obtain precise estimations of MAPSE, performing coordinate extraction, rotation correction, peak detection and MAPSE calculation. The TEE B-mode recordings were acquired at St. Olavs University Hospital and made available for use in this study by close collaborators at the Department of Circulation and Imaging at NTNU. The landmark detector demonstrated that detecting mitral landmarks using deep learning was feasible for TEE B-mode images.

After the proposed pipeline was completed, the clinically obtained MAPSE values supplied by Dr. Rye Berg were applied for a final evaluation of the pipeline. The results, presented in section 4.3, show that the pipeline is able to accurately estimate MAPSE when evaluated against clinically obtained values, with a mean error of -0.08 ± 1.38. Certain outliers are produced, but the pipeline does not produce systematic errors. With further development of the ideas and methods presented in this thesis, the accuracy and robustness of the MAPSE detections will be improved.

This project has been the first study at NTNU aiming to automatically detect MAPSE in TEE B-mode recordings using deep learning, and the results show great promise for future utilization of automatic MAPSE detection.

References

[1] T. Weiser, A. Haynes, G. Molina, S. R Lipsitz, M. M Esquivel, T. Uribe-Leitz, R. Fu, T. Azad, T. Chao, W. Berry, and A. A Gawande, "Size and distribution of the global volume of surgery in 2012," *Bulletin of the World Health Organization*, vol. 94, pp. 201–209F, 03 2016.

[2] A. Rafiq, E. Sklyar, and J. N. Bella, "Cardiac evaluation and monitoring of patients undergoing noncardiac surgery," in *Health services insights*, 2017.

[3] L. A. Fleisher, J. A. Beckman, K. A. Brown, H. Calkins, E. L. Chaikof, K. E. Fleischmann, W. K. Freeman, J. B. Froehlich, E. K. Kasper, J. R. Kersten, B. Riegel, and J. F. Robb, "Acc/aha 2007 guidelines on perioperative cardiovascular evaluation and care for noncardiac surgery," *Circulation*, vol. 116, no. 17, pp. e418–e500, 2007.

[4] V. et al., "Perioperative cardiovascular monitoring of high-risk patients: a consensus of 12," *Critical Care*, vol. 19, p. 224, 2015.

[5] A. Mohamed, A. Arifi, and A. Omran, "The basics of echocardiography," *Journal of the Saudi Heart Association*, vol. 22, pp. 71–6, 04 2010.

[6] S. T. Reeves, A. C. Finley, N. J. Skubas, M. Swaminathan, W. S. Whitley, K. E. Glas, R. T. Hahn, J. S. Shanewise, M. S. Adams, and S. K. Shernan, "Basic perioperative transesophageal echocardiography examination: A consensus statement of the american society of echocardiography and the society of cardiovascular anesthesiologists," *Journal of the American Society of Echocardiography*, vol. 26, no. 5, pp. 443 – 456, 2013.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[8] S. Duffner and C. Garcia, "A connexionist approach for robust and precise facial feature detection in complex scenes," pp. 316 – 321, 10 2005.

[9] M. Bodini, "A review of facial landmark extraction in 2d images and videos using deep learning," *Big Data and Cognitive Computing*, vol. 3, p. 14, 02 2019.

[10] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3476–3483, June 2013.

[11] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin, "Extensive facial landmark localization with coarse-to-fine convolutional network cascade," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[12] D. Merget, M. Rock, and G. Rigoll, "Robust facial landmark detection via a fully-convolutional local-global context network," pp. 781–790, 06 2018.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[14] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, pp. 749–753, May 2018.

[15] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016* (S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, eds.), (Cham), pp. 424–432, Springer International Publishing, 2016.

[16] A. Jansson, E. J. Humphrey, N. Montecchio, R. M. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep u-net convolutional networks," in *ISMIR*, 2017.

[17] F. T. Dezaki, N. Dhungel, A. H. Abdi, C. Luong, T. Tsang, J. Jue, K. Gin, D. Hawley, R. Rohling, and P. Abolmaesumi, "Deep residual recurrent neural networks for characterisation of cardiac cycle phase from echocardiograms," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (M. J. Cardoso, T. Arbel, G. Carneiro, T. Syeda-Mahmood, J. M. R. Tavares, M. Moradi, A. Bradley, H. Greenspan, J. P. Papa, A. Madabhushi, J. C. Nascimento, J. S. Cardoso, V. Belagiannis, and Z. Lu, eds.), (Cham), pp. 100–108, Springer International Publishing, 2017.

[18] G. Carneiro and J. C. Nascimento, "Combining multiple dynamic models and deep learning architectures for tracking the left ventricle endocardium in ultrasound data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 2592–2607, Nov 2013.

[19] M. Sofka, F. Milletari, J. Jia, and A. Rothberg, "Fully convolutional regression network for accurate detection of measurement points," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (M. J. Cardoso, T. Arbel, G. Carneiro, T. Syeda-Mahmood, J. M. R. Tavares, M. Moradi, A. Bradley, H. Greenspan, J. P. Papa, A. Madabhushi, J. C. Nascimento, J. S. Cardoso, V. Belagiannis, and Z. Lu, eds.), (Cham), pp. 258–266, Springer International Publishing, 2017.

[20] K. Isaaz, A. Thompson, G. Ethevenot, J. L. Cloez, B. Brembilla, and C. Pernot, "Doppler echocardiographic measurement of low velocity motion of the left ventricular posterior wall," *The American Journal of Cardiology*, vol. 64, no. 1, pp. 66 – 75, 1989.

[21] A. Heimdal, A. Støylen, H. Torp, and T. Skjærpe, "Real-time strain rate imaging of the left ventricle by ultrasound," *Journal of the American Society of Echocardiography*, vol. 11, no. 11, pp. 1013 – 1019, 1998.

[22] D. Pellerin, R. Sharma, P. Elliott, and C. Veyrat, "Tissue doppler, strain, and strain rate echocardiography for the assessment of left and right systolic ventricular function," *Heart*, vol. 89, no. suppl 3, pp. iii9–iii17, 2003.

[23] K. K. Kadappu and L. Thomas, "Tissue doppler imaging in echocardiography: Value and limitations," *Heart, Lung and Circulation*, vol. 24, no. 3, pp. 224 – 233, 2015.

[24] H. Blessberger and T. Binder, "Two dimensional speckle tracking echocardiography: basic principles," *Heart*, vol. 96, no. 9, pp. 716–722, 2010.

[25] M. Leitman, P. Lysyansky, S. Sidenko, V. Shir, E. Peleg, M. Binenbaum, E. Kaluski, R. Krakover, and Z. Vered, "Two-dimensional strain–a novel software for real-time quantitative echocardiographic assessment of myocardial function," *Journal of the American Society of Echocardiography*, vol. 17, no. 10, pp. 1021 – 1029, 2004.

[26] S. A. Reisner, P. Lysyansky, Y. Agmon, D. Mutlak, J. Lessick, and Z. Friedman, "Global longitudinal strain: a novel index of left ventricular systolic function," *Journal of the American Society of Echocardiography*, vol. 17, no. 6, pp. 630 – 633, 2004.

[27] M. H. Hassel, "Perioperative monitoring of cardiac function based on transesophageal echocardiographic data," Master's thesis, NTNU, 6 2018.

[28] M. van Stralen, K. Leung, M. Voormolen, N. de Jong, A. van der Steen, J. Reiber, and J. Bosch, "Time continuous detection of the left ventricular

long axis and the mitral valve plane in 3-d echocardiography," *Ultrasound in Medicine & Biology*, vol. 34, no. 2, pp. 196 – 207, 2008.

[29] H. De Veene, P. B. Bertrand, N. Popovic, P. M. Vandervoort, P. Claus, M. De Beule, and B. Heyde, "Automatic mitral annulus tracking in volumetric ultrasound using non-rigid image registration," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1985–1988, Aug 2015.

[30] E. Smistad, A. Østvik, I. Mjal Salte, S. Leclerc, O. Bernard, and L. Lovstakken, "Fully automatic real-time ejection fraction and mapse measurements in 2d echocardiography using deep neural networks," in *2018 IEEE International Ultrasonics Symposium (IUS)*, pp. 1–4, Oct 2018.

[31] J. F. Grue, S. Storve, H. Dalen, Øyvind Salvesen, O. C. Mjølstad, S. O.Samstad, H. Torp, and B. O. Haugen, "Automatic measurements of mitral annular plane systolic excursion and velocities to detect left ventricular dysfunction," *Ultrasound in Biology and Medicine*, vol. 44, pp. 168–176, 2018.

[32] OpenStax, *Anatomy and Physiology*. OpenStax CNX, 4 2019. `http://cnx.org/contents/14fb4ad7-39a1-4eee-ab6e-3ef2482e3e22@15.2`.

[33] DestinyQx, "Wiggers diagram," 2019. `https://commons.wikimedia.org/wiki/File:Wiggers_Diagram_2.svg`.

[34] A. Shalbaf, H. Behnam, Z. Alizade-Sani, and M. Shojaeifard, "Automatic classification of left ventricular regional wall motion abnormalities in echocardiography images using nonrigid image registration," *Journal of digital imaging : the official journal of the Society for Computer Applications in Radiology*, vol. 26, 01 2013.

[35] L. Bergenzaun, H. Öhlin, P. Gudmundsson, R. Willenheimer, and M. S. Chew, "Mitral annular plane systolic excursion (mapse) in shock: a valuable echocardiographic parameter in intensive care patients," *Cardiovascular Ultrasound*, vol. 11, p. 16, May 2013.

[36] A. Støylen, H. E. Mølmen, and H. Dalen, "Relation between mitral annular plane systolic excursion and global longitudinal strain in normal subjects: The hunt study," *Echocardiography*, vol. 35, no. 5, pp. 603–610, 2018.

[37] D. P. Borde, S. Joshi, B. Asegaonkar, P. Apsingkar, S. Pande, S. More, U. Takalkar, and A. Deodhar, "Mitral annular plane systolic excursion: A simple, reliable echocardiographic parameter to detect left ventricular systolic dysfunction in patients undergoing off-pump coronary artery bypass grafting

with transesophageal echocardiography," *Journal of Cardiothoracic and Vascular Anesthesia*, vol. 33, no. 5, pp. 1334 – 1339, 2019.

[38] P. Hoskins, K. Martin, and A. Thrush, *Diagnostic Ultrasound Physics and Equipment*. Cambridge University Press, 2010.

[39] V. Chan and A. Perlas, *Basics of Ultrasound Imaging*, pp. 13–19. New York, NY: Springer New York, 2011.

[40] H. Kasban, M. El-bendary, and D. Salama, "A comparative study of medical imaging techniques," *International Journal of Information Science and Intelligent System*, vol. 4, pp. 37–58, 04 2015.

[41] StayWellCompany, "Tee," 2019. `https://mychart.geisinger.org/staywel/html/Images/Image_201510071501_2087.jpg`.

[42] W. G. Daniel and A. Mügge, "Transesophageal echocardiography," *New England Journal of Medicine*, vol. 332, no. 19, pp. 1268–1280, 1995. PMID: 7708072.

[43] R. T. Hahn, T. Abraham, M. S. Adams, C. J. Bruce, K. E. Glas, R. M. Lang, S. T. Reeves, J. S. Shanewise, S. C. Siu, W. Stewart, and M. H. Picard, "Guidelines for performing a comprehensive transesophageal echocardiographic examination: Recommendations from the american society of echocardiography and the society of cardiovascular anesthesiologists," *Journal of the American Society of Echocardiography*, vol. 26, no. 9, pp. 921 – 964, 2013.

[44] C. Otto, *Textbook of Clinical Echocardiography*. Elsevier, 2013.

[45] P. S. Crowther and R. J. Cox, "A method for optimal division of data sets for use in neural networks," in *Knowledge-Based Intelligent Information and Engineering Systems* (R. Khosla, R. J. Howlett, and L. C. Jain, eds.), (Berlin, Heidelberg), pp. 1–7, Springer Berlin Heidelberg, 2005.

[46] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[47] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain [j]," *Psychol. Review*, vol. 65, pp. 386 – 408, 12 1958.

[48] S. University, "Biological neuron." `https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html`.

[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[50] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

[51] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 2146–2153, 2009.

[52] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.

[53] S. Magazine, "Nonconvex optimization." `https://towardsdatascience.com/its-only-natural-an-excessively-deep-dive-into-natural-gradient-optimization-`

[54] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, July 2011.

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[56] T. Nordal, "Deep learning based tracking of anatomical structures from transesophageal recordings of the left ventricle," Master's thesis, NTNU, 12 2018.

[57] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv:1803.01271*, 2018.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[59] F. Milletari, N. Navab, and S. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," *CoRR*, vol. abs/1606.04797, 2016.

[60] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[62] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016.

[63] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv e-prints*, p. arXiv:1603.07285, Mar 2016.

[64] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *CoRR*, vol. abs/1409.0575, 2014.

[65] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv e-prints*, p. arXiv:1409.1556, Sep 2014.

[66] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *arXiv e-prints*, p. arXiv:1411.4038, Nov 2014.

[67] D. Masters and C. Luschi, "Revisiting Small Batch Training for Deep Neural Networks," *arXiv e-prints*, p. arXiv:1804.07612, Apr 2018.

[68] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv e-prints*, p. arXiv:1608.06993, Aug 2016.

[69] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *arXiv e-prints*, p. arXiv:1512.00567, Dec 2015.

[70] R. O. Bonow, S. L. Bacharach, C. Crawford-Green, and M. V. Green, "Influence of temporal smoothing on quantitation of left ventricular function by gated blood pool scintigraphy," *The American Journal of Cardiology*, vol. 64, no. 14, pp. 921 – 925, 1989.

[71] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell, "Characterizing and Avoiding Negative Transfer," *arXiv e-prints*, p. arXiv:1811.09751, Nov 2018.

Appendix A

Submitted Abstract for EuroEcho19

Automated detection of mitral annular plane systolic excursion in transoesophageal echocardiography based on deep learning.

Introduction: Major surgery and interventions may impact cardiac function. Perioperative monitoring is currently based on vital signs and clinical observations. However, this does not offer a complete monitoring of left ventricular function throughout the intervention. We hypothesize that functional monitoring of the heart can be performed automatically based on transoesphageal echocardiography (TOE) images. One parameter that has been shown to correlate well with ejection fraction is mitral annular plane systolic excursion (MAPSE). To aid functional monitoring of the left ventricle perioperatively, we propose a technique for detecting MAPSE in TOE images of the left ventricle.

Purpose: The purpose of this study is to automatically track the movement of the mitral annular plane in TOE sequences of the left ventricle and detect MAPSE via a deep learning approach.

Method: Recordings from 131 consecutive complete TOE exams from the Echocardiography Unit were anonymized and used for training. Recordings from 23 consecutive TOE exams, also anonymized, were used as test set. All recordings were manually annotated with the location of the landmarks indicated in both 4-chamber (4C) and 2-chamber (2C) views. All recordings were made using state-of-the-art clinical scanners. The captures include 3 to 5 heart cycles of standard 4C and 2C views.

An approach based on a fully convolutional neural network was implemented and

trained in a supervised manner to predict the location of two landmarks on the mitral annular plane in B-mode TOE images from 4C and 2C views. The model was also trained to account for noise by recognizing when detecting the landmarks is not feasible due to poor image quality. We have implemented all necessary post processing calculations to automatically estimate MAPSE based only on raw TOE B-mode sequences.

Results: Preliminary results on the test data show that the landmark detector is able to track the vertical movement of landmarks on the mitral annular plane with a mean absolute error of 0.88 mm and a standard deviation of 0.27 mm (Fig. 1: Upper left and lower left: tracked mitral attachment points on a sample case presented upper right. Lower right: all measured Y-axis excursion values versus the reference). The classifier for detecting ultrasound frames where landmark detection is not feasible has a sensitivity of 0.82 and a specificity of 0.91.

Conclusion: The landmark detector is showing promising results in tracking of the mitral annular plane excursion. This can provide a fast calculation of MAPSE and eliminate intraobserver variability. This may be included in a more extensive cardiac monitoring for any type of surgery without the need of manual input from echocardiographers. Further research is ongoing and a comparison with clinical MAPSE values is underway.

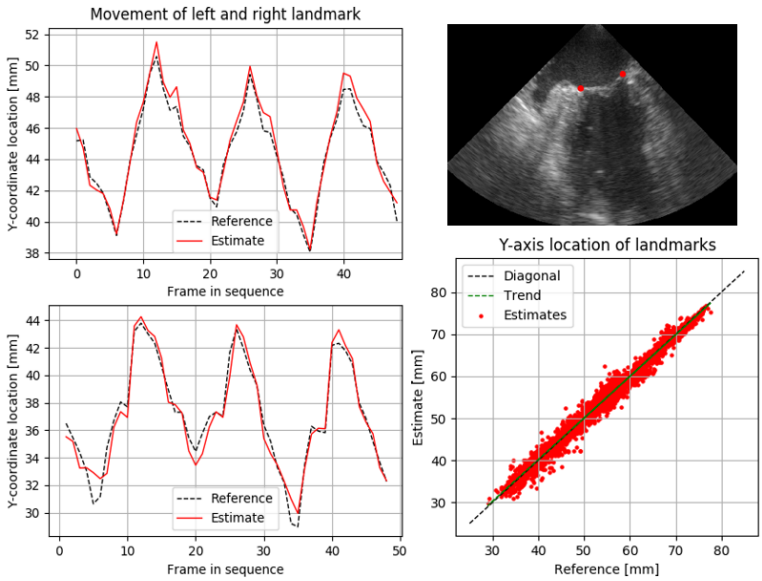
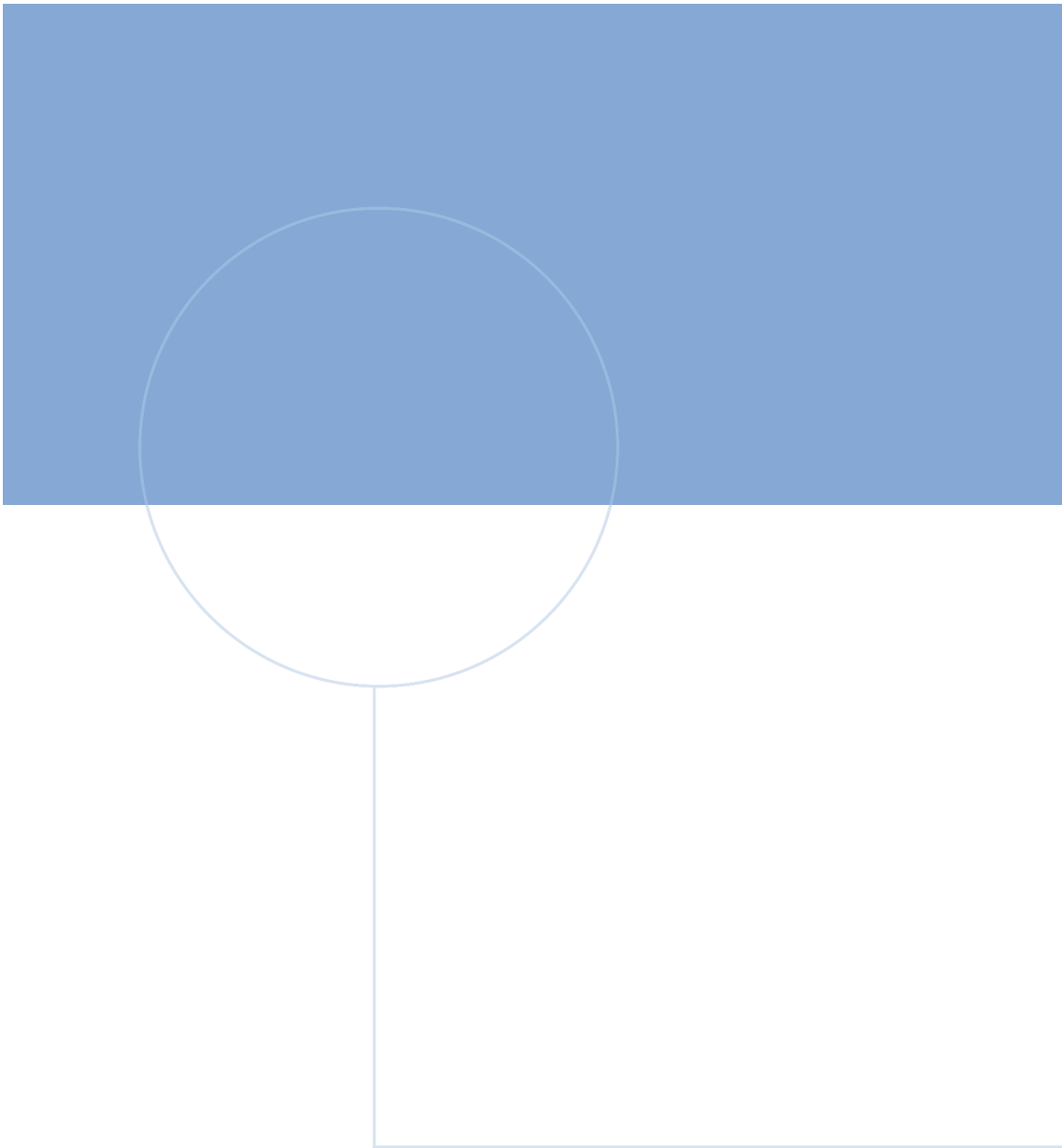


Fig 1



NTNU
Norwegian University of
Science and Technology