NTNU
Norwegian University of
Science and Technology

Segmentation of Coronary Arteries from CT-scans of the heart using Deep Learning

Øyvind Kjerland

Master of Science in Computer Science
Submission date: June 2017
Supervisor: Frank Lindseth, IDI

Norwegian University of Science and Technology
Department of Computer Science

Abstract

Image segmentation is an important tool in several fields. One is medical image computing where the images are divided into regions based on tissue type and organ, which can further be used for visualization and diagnosis. Due to the large amount of data produced by modern imaging modalities such as CT and MRI, the process of manual or semi-automatic segmentation is time consuming, tedious and introduces bias by clinical experts. Recent advances in the field of deep learning has given rise to several fully automatic methods for robust segmentation on a large variety of segmentation tasks. In this thesis the use of a 3D convolutional neural network architecture is used on three different segmentation tasks is explored. These tasks are coronary artery segmentation, brain tumor segmentation and digital rock segmentation.

Coronary artery disease is the leading cause of death in Europe. Diagnosis of the disease is today done by invasive methods, but research on using computational fluid dynamics to model the blood flow based on non-invasive imaging show great promise. In this thesis a method for fully automatic segmentation of the coronary arteries based on deep learning is proposed, implemented and evaluated on a dataset provided by St. Olavs Hospital. The dataset contains manual segmentations performed by a clinical expert. The proposed method uses two neural networks trained on aorta segmentation and coronary segmentation respectively and is able to segment the complete coronary artery tree in some test images, but fails to segment all branches in the rest of the images.

For the brain tumor segmentation task a network is trained and evaluated on a dataset provided by the Norwegian National Advisory Unit for Ultrasound and Image Guided Therapy (USIGT). The results show that the deep learning method is able to produce good segmentations fully automatically. These segmentations do however inlcude some spurious responses.

Digital Rocks technology is based on using high resolution 3D microscopy imaging to create models describing reservoir rock. A network was trained and evaluated on a digital rock dataset provided by FEI. The results show that the network is able to produce good segmentations fully automatically.

Sammendrag

Bildesegmentering er et viktig verktøy innenfor flere felt. Et av disse er medical image computing, hvor bilder er delt inn i regioner basert på vevstype og organer, som kan videre bli brukt for visualisering eller diagnosering. På grunn av den store mengden data som blir produsert av moderne avbildningsmetoder som CT og MRI er prosessen for manuell og semi-automatisk segmentering tidkrevende, langtekkelig og introduserer bias fra kliniske eksperter. Nylige fremskritt innenfor feltet dyp læring har gitt opphav til flere helautomatiske metoder for robust segmentering på en rekke segmenteringsoppgaver. I denne oppgaven er bruk av en 3D convolutional nevralt nettverk arkitektur på tre ulike segmenteringsoppgaver utforsket. Disse oppgavene er segmentering av koronar arterier, segmentering av hjernesvulst og segmentering av digital rock.

Koronararteriesykdom er den ledende dødsårsaken i Europa. Diagnose av sykdommen er i dag gjennomført ved bruk av invasive metoder, men forskning på bruk av numerisk fluiddynamikk for å modellere blodstømningen basert på ikke-invasive avbildningsmetoder viser lovende resultat. I denne oppgaven er en metode for helautomatisk segmentering av koronararteriene basert på dyp læring foreslått, implementert og evaluert på dataset gitt av St. Olavs Hospital. Datasettet inneholder manuelle segmenteringer utført av en klinisk ekspert. Den foreslåtte metoden bruker to nevrale nettverk trent på henholdsvis aorta segmentering og koronararterie segmentering og er i stand til å segmentere hele koronararterietreet i noen av bildene, men feiler å segmentere alle grenene i resten av bildene.

For segmentering av hjernesvulst er et nettverk trent og evaluert på et dataset fra nasjonal kompetansetjeneste for ultralyd of bildeveiledet behandling (USIGT). Resultatene viser at metoden basert på dyp læring greier å produsere gode segmenteringer. Segmenteringene inneholder allikevel noen falske responser.

Digital Rocks teknologi er basert på bruk av høyoppløselig 3D mikroskopi for å lage modeller som beskriver reservoarberg. Et nevralt nettverk er trent og evaluert på et digital rock dataset fra FEI. Resultatene viser at nettverket gir gode segmenteringer helautomatisk.

Table of Contents

List of Tables

List of Figures

Abbreviations

CAD	=	Coronary Artery Disease
IHD	=	Ischemic Heart Disease
RCA	=	Right Coronary Artery
LCA	=	Left Coronary Artery
LCX	=	Left Circumflex Coronary Artery
LAD	=	Left Anterior Descending Coronary Artery
GBM	=	Glioblastoma Multiforme
CT	=	Computed Tomography
CTA	=	Computed Tomograpgy Angiogram
MRI	=	Magnetic Resonance Imaging
ANN	=	Artificial Neural Network
ReLU	=	Rectified Linear Unit
CNN	=	Convolutional Neural Network
FM	=	Feature Map
GPU	=	Graphical Processing Unit
ROI	=	Region Of Interest
BT	=	Bentheimer sandstone
BR	=	Berea sandstone
CA	=	Carbonate
DSC	=	Dice Similarity Coefficient

Chapter 1

Introduction

Image segmentation is process of dividing an image into smaller partitions based on some meaningful characteristics of the pixels in the image. In terms of medical imaging, an image can be divided into regions which can be certain tissue types, organs or other relevant structures (45). The segmentations are used for quantitative analysis and diagnosis. The gold standard in medical image segmentation is manual segmentations performed by clinical experts. This is a time consuming task as modern medical imaging modalities, such as CT and MRI, are able to produce a large amount of data in the form of 3D image volumes. The manual segmentations are also subject to bias and human error. Several semi-automatic methods have been used to speed up the segmentation process but still requires a clinical expert to initialize and/or guide the segmentation. The importance of fully automatic segmentation methods increases with the increasing amount of data available, both individually for each patient and for all patients as a whole. Fully automatic segmentation is also important in domains other than medical imaging, such as industrial applications where imaging is used for analysis and validation. In this thesis three segmentation tasks are explored: coronary artery segmentation, brain tumor segmentation and digital rocks segmentation.

Coronary artery disease (CAD) is the leading cause of death among both men and women in Europe (36). Diagnosis of the disease is today done by invasive methods. Research in computational fluid dynamics shows promising results on modelling the coronary arteries from non-invasive imaging. However, the process of obtaining these models needs to be done either manually or in a semi-automatic fashion by experts and is time consuming. A fully automatic method for extracting the coronary arteries is therefore highly desirable.

In this thesis, a method for automatic segmentation of the coronary arteries using deep learning is proposed. The method uses the DeepMedic architecture by Kamnitsas et al. (21) as a component. To improve the output of the deep learning method, a combination of multiple networks trained on aorta segmentation and coronary artery segmentation are used to further refine the result, inspired by related work on coronary artery segmentation.

The networks are trained on data provided by the Rotterdam Coronary Artery Algorithm Evaluation Framework (41; 22). To evaluate the proposed method, it is tested on a dataset provided by St. Olavs Hospital that has been manually segmented by a clinical expert.

The DeepMedic architecture was originally used for brain lesion and tumor segmentation. With the purpose of seeing how well the network performed on a similar task, the method was trained and validated on a brain tumor datset provided by USIGT (12). This dataset contains manual segmentations of glioblastoma, a type of brain tumor, performed by two clinical experts. A fully automatic segmentation method is desireable as the manual segmentation is a time consuming process and requires clinical experts.

The DeepMedic architecture was further evaluated on a third segmentation task which was segmentation of rock samples from micro-CT volumes. The dataset was provided by FEI (1). By segmenting different parts of the rock volumes, such as pore and grain, it is possible to use the segmentation output to calculate macroscopic properties of the rock.

1.1 Research Questions

The overall goal of this thesis is to explore the use of deep learning for image segmentation performed on 3D image volumes and implement a method for fully automatic segmentation of the coronary arteries. This gives the following research questions:

1. Can deep learning be used to create a fully automatic segmentation method for coronary segmentation?
2. Is it possible to use the same type of deep learning method to perform fully automatic segmentation of brain tumors from MRI volumes?
3. Can the same deep learning method be trained perform segmentation on digital rocks.

1.2 Structure

In this section an overview of the structure of this thesis is presented.

1.2.1 Introduction

In this chapter the introduction and motivation for the thesis are presented.

1.2.2 Background

This chapter provides the basic theory for the methods used in this thesis and a literature review of related work. The basic theory first provides a summary of the application domains used in this thesis, which are coronary arteries, brain tumors and digital rocks. This is followed by the theory behind the different imaging modalities used, which are CT

and MRI. A brief explanation of the basic theory behind deep learning using convolutional neural networks is then given. For the literature review the previous work related to this thesis is presented and discussed. Methods related to coronary artery segmentation are first presented. This is followed by a review of methods for medical image segmentation using deep learning. Lastly, deep learning methods used for coronary artery segmentation and brain tumor segmentation are reviewed.

1.2.3 Method

In this chapter the chosen method is presented. The chosen deep learning method is explained, followed by a summary of the different datasets that are used in this thesis. A section is used to describe the preprocessing steps applied to the datasets. The different experiments that were set up are then explained, first experiments for coronary artery segmentation, then brain tumor segmentation and segmentation on the digital rocks dataset. Finally the metrics used to evaluate the methods are presented.

1.2.4 Results

The results of the experiments are presented and compared. Qualitative results are based on visual comparisons of input data, corresponding ground truth data and the output of the methods. Quantitative results are presented in the form of evaluation on different metrics as well as results from the evaluation frameworks.

1.2.5 Discussion

In this chapter the results are discussed. The strengths and weaknesses of the chosen method are discussed based on the results on coronary artery segmentation, brain tumor segmentation and segmentation on the rock dataset.

1.2.6 Conclusion and future work

A conclusion is made from the results, the following discussion and the initial research questions. Future work based on the discussion in the previous chapter is then presented.

Chapter 2

Background

This chapter provides the basic theory behind the methods used in this thesis and a literature review of related work.

2.1 Basic Theory

The basic theory first provides a summary of the application domains used in this thesis, which are coronary arteries, brain tumors and digital rocks. This is followed by the theory behind the different imaging modalities used, which are CT and MRI. A brief explanation of the basic theory behind deep learning using convolutional neural networks is then presented.

2.1.1 Application domains

In this thesis segmentation tasks on several domains are presented. These are coronary arteries, brain tumors and digital rocks.

Coronary Artery Disease

The coronary arteries are the blood vessels located around the heart providing the myocardium with blood (9). The right coronary artery (RCA) and left coronary artery (LCA) originate from the aorta just above where it exits the left ventricular chamber. The LCA further branches into the left circumflex coronary artery (LCX) and the left anterior descending coronary artery (LAD). Figure 2.1 shows an overview over where the coronary arteries are located in relation to the heart.

Coronary artery disease (CAD), also called Ischemic heart disease (IHD), is a disease that can lead to angina (chest pain), myocardial infarction (heart attack) and cardiac arrest (35). CAD is the leading cause of death globally. CAD is caused by plaque building up over time. The plaque is made up of fatty lipid deposits, cholesterol, calcium and other

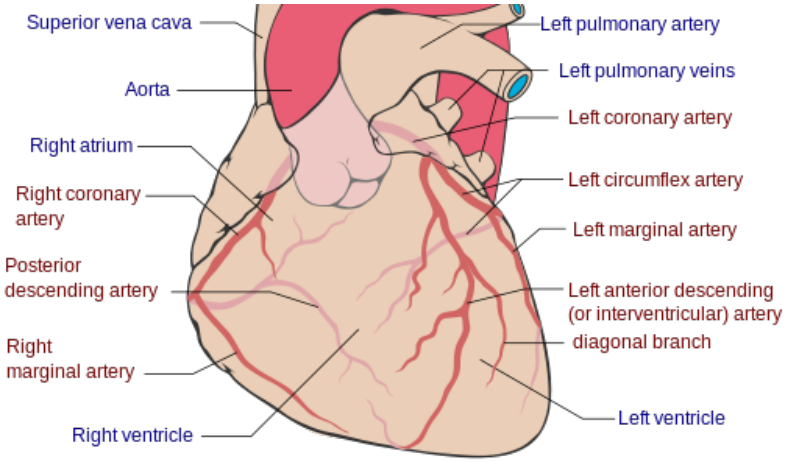


Figure 2.1: Diagram of the coronary arteries, By Coronary.pdf: Patrick J. Lynch, medical illustratorderivative work: Fred the Oyster (talk)adaption and further labeling: Mikael Häggström - Coronary.pdf, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=9967381

substances found in the bloodstream. These deposits causes the arteries to narrow and harden, also known as atherosclerosis. Figure 2.2 illustrates this. The narrowing causes the heart to not receive enough oxygen-rich blood, which leads to angina and myocardial infarction. Severly narrowed parts (stenoses) can cause blood clots which can lead to cardiac arrest do to sudden lack of oxygen provided to the heart muscle (35). Risk factors include genetics, diabetes, smoking, obesity, lack of exercise and stress.

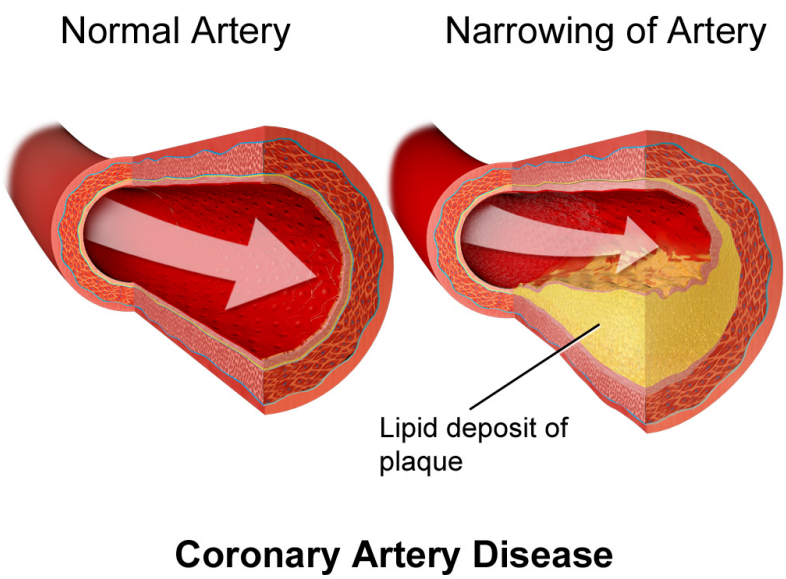


Figure 2.2: Coronary artery disease caused by buildup of plaque. Blausen.com staff (2014). 'Medical gallery of Blausen Medical 2014'. WikiJournal of Medicine 1 (2). DOI:10.15347/wjm/2014.010. ISSN 2002-4436. [CC BY 3.0 (http://creativecommons.org/licenses/by/3.0)], via Wikimedia Commons

Diagnosis can be performed in several ways (3). Echocardiogram is based on using ultrasound to monitor the heartbeats. If irregular heart motion is observed it could be a symptom that the heart is not getting enough blood caused by CAD. Another method is the use of Computed Tomography Angiogram (CTA). This can also be used in a combination with invasive coronary angiogram, where a catheter is inserted into the proximal parts of the coronary arteries and contrast fluid is then injected from the catheter. By doing this the coronary arteries are easier to observe and it makes it possible to observe the blood flow to easier detect narrowed vessels.

Treatment includes lifestyle changes, medication and in some cases angioplasty and bypass surgeries (3). In angioplasty a small catheter with a balloon-like stent is inserted into into the stenosed region to remove the plaque. With bypass surgeries a new artery is extended from the aorta and surgically inserted past the blocked passage.

Brain tumors

There exists several types of brain tumors, where glioblastoma is one of these types. Also called glioblastoma multiforme (GBM), it makes up 15% of all brain tumor cases and is regarded as the most agressive type of cancer beginning in the brain (53; 5). Symptoms are similar to that of a stroke. Genetic disorders can lead to increased risk of suffering from glioblastoma, but in most of the cases the direct cause is unknown (13). Diagnosis is usu-

ally performed by a neurological exam followed by a MRI scan and/or tissue biopsy (53). Common treatment consists of surgery with removal of the tumor, followed by radiation therapy and chemotherapy (13). Typical life expectancy after diagnosis is between 12 and 15 months (13). Figure 2.3 show how glioblastoma can look on a T1 weighted MRI scan. The image is a slice of the first volume in the dataset provided by the Norwegian National Advisory Unit for Ultrasound and Image Guided Therapy (USIGT) (12).

Digital Rocks

Digital Rocks technology is based on using high resolution 3D microscopy imaging to create models describing reservoir rock (1). By using these models together with simulation tools, macroscopic rock properties can be obtained. The quality of these models relies on accurate segmentations, obtained by performing image processing on the 3D volumes. Each voxel in the segmented result is labelled based on their phase, grain, micro-phase or pore.

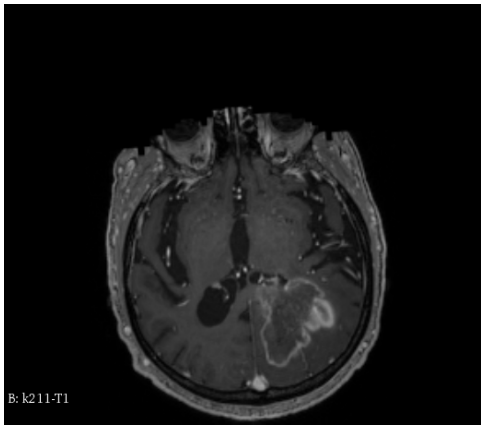


Figure 2.3: Example of globlastoma, here shown as a non uniform region with a bright border towards the lower right of the image.

2.1.2 Imaging modalities

Data obtained from several different imaging modalities are used in this thesis. The data used for coronary artery segmentation is in the form of CT scans. The brain tumor dataset consists of MRI volumes. The digital rock dataset uses a form of X-Ray tomography for 3D microscopy.

CT

Rubin et al.(40) provides a thorough review of the application of computed tomography (CT) angiography for diagnosis of CAD. The most common form of CT scans are based on X-ray imaging. X-rays are produced by firing an electron beam at a heavy metal, usually

tungsten. When the electrons hit the atoms of the metal, they can either be slowed down or they can cause electrons to jump to a higher energy state. When these electrons go back to their original energy state, they release energy in form of photons (40). By adjusting the energy of the electron beam it is possible to make this process generate x-rays. The x-ray producer is then aimed at a silver halide film. The patient is then placed between the x-ray generator and the film. Different tissues found in the body will attenuate the x-rays differently. Air will attenuate the least, soft tissue will attenuate some of the x-rays, while bone and teeth will attenuate everything. When the x-rays hit the silver halide, it darkens the film, which is why the lungs will show up as nearly black on x-ray images and bone will be almost completely white. The imaging is not without risk, as x-rays can damage cells and DNA, which can lead to development of cancer (40).

The problem with common X-ray imaging is that they are a planar projection of a 3D volume, which means we get occlusions caused by bones and it is generally hard to observe the depth. CT imaging solves this issue by taking images from multiple angles (40). For each angle, a one dimensional image will be stored which is obtained by using a fan-shaped X-ray beam going through the patient. By taking enough of these fan-shaped images, it is possible to use tomography to reconstruct the original cross section. This is usually done by filtered forward-back-projection, where each ray from the one dimensional images are filtered and projected back onto a plane at the given angle. After all the rays has been projected back, the image is normalized and the reconstructed tomography is obtained. By combining several slices it is possible to reconstruct a 3D volume of the patient.

MRI

Magnetic resonance imaging (MRI) is a technique that does not use x-rays. To generate an image strong magnetic fields and radio waves are used. The technology is based on nuclear magnetic resonance. When put under a strong magnetic field, the nuclear spin of the hydrogen atoms found in the body (water, fat) will align with the field (20). By sending a pulse sequence of radio waves it is possible to excite the nuclear spin into a different energy state. When the pulse is turned off the atoms will go back to their original aligned state, releasing radio waves in the process that are received by antennas. Hydrogen atoms in different tissues will have different relaxation properties. By using different pulse sequences it is possible to differentiate between the different tissues. To localize the response a magnetic field gradient in three dimensions is used. The advantages over CT images is that MRI can essentially observe the tissue type at any position without degrading the quality from attenuation of bones and other dense tissues. Another advantage is that there are no health risks similar to the effect of ionizing x-rays produced by CT scanners. Compared to obtaining a CT scan, MRI is a much slower process and requires the patient to stay still for a long period of time. This also makes it more expensive to use MRI imaging. Due to using a strong magnetic field, people with implants made of magnetic material, such as pacemakers, will be unable to use MRI imaging (20).

2.1.3 Deep Learning

Deep learning is a form of machine learning dealing with artificial neural networks (ANN) containing mutliple layers. The Deep Learning Book (14) contains a thorough explanation of deep learning methods. In this section the basics of ANN are explained, starting with the building blocks and concepts and leading up to convolutional neural networks (CNN) performed on images.

Artificial neural network

An ANN is a method that takes a number of inputs, such as pixel values in an image, and produces some output, for example a prediction of which organ can be seen in the image. Figure 2.4 shows a highly simplified overview of using a network to classify an image as a certain type of organ.

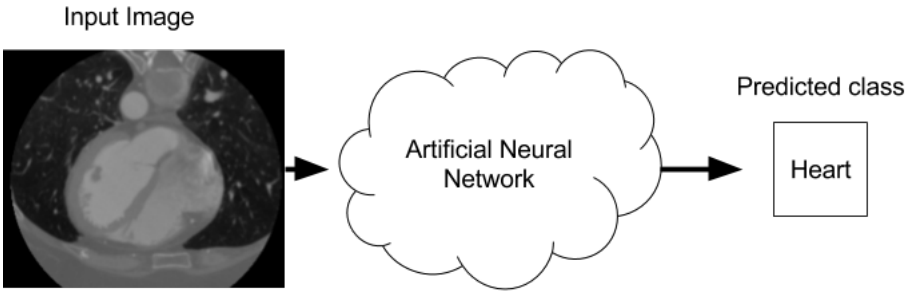


Figure 2.4: Example usecase of an artificial neural network. Here a network has been trained to classify the input image.

A more detailed explanation of ANN is that they consist of two or more layers that are connected together, where each connection has a learnable weight parameter. Figure 2.5 shows a simple feed-forward network with one input layer, two hidden layers and an output layer. Each layer consists of multiple neurons that receive input from all neurons in the previous layer and sends output to all the neurons in the next layer. This is called a fully connected layer. The ANN can have any number of hidden layers and each layer can have any number of neurons.

Neuron

The input for a neuron in layer l is the sum of the outputs $\mathbf{y}_{l-1}$ of the previous layer multiplied by the weights $\mathbf{w}_l^i$, where i is the index of the neuron. To obtain the output a bias b_l^i is added and a non-linear activation function is used on the result to produce the output y_l^i. Figure 2.6 illustrates the process. This can be represented by equation 2.1,

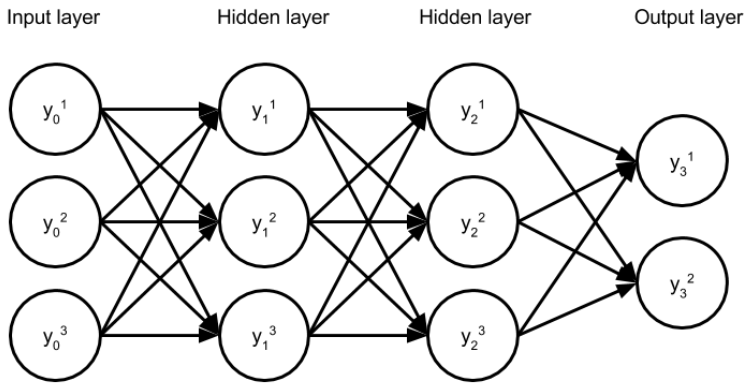


Figure 2.5: Example of a simple fully connected neural network consisting of an input layer, two hidden layers and an output layer.

where N_l is the number of neurons in layer l.

$$y_l^i = \sigma\left(\sum_{j=1}^{N_{l-1}} y_{l-1}^j w_l^{j,i} + b_l^i\right) \tag{2.1}$$

The summation in equation 2.1 can also be represented as the dot product of the vector $\mathbf{y}_{l-1}$ and $\mathbf{w}_l^i$, giving equation 2.2.

$$y_l^i = \sigma\left((\mathbf{y}_{l-1})^T \mathbf{w}_l^i + b_l^i\right) \tag{2.2}$$

The output vector of layer l can then be represented by equation 2.3, where $\mathbf{W}_l$ is a N_{l-1}, N_l matrix.

$$\mathbf{y}_l = \sigma\left((\mathbf{y}_{l-1})^T \mathbf{W}_l + \mathbf{b}_l\right) \tag{2.3}$$

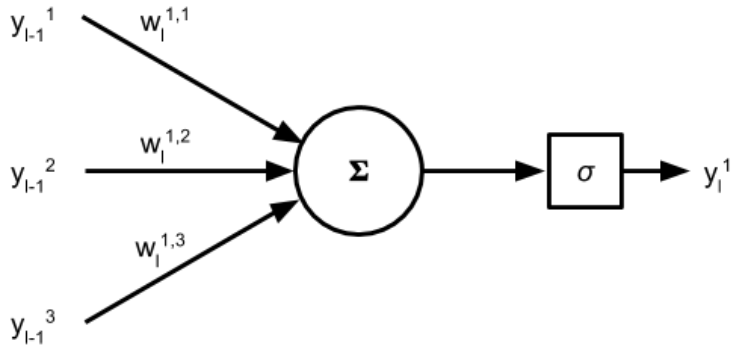


Figure 2.6: Example of a neuron.

Activation function

Possible choices for the activation function σ includes sigmoid, tanh, ReLU and others. Most modern architectures use some variation of the Rectifying linear unit (ReLU) activation function. The ReLU is defined by equation 2.4.

$$\sigma(x) = \begin{cases} 0 & : x < 0 \\ x & : x \geq 0 \end{cases} \tag{2.4}$$

A version of the ReLU is the PReLU (17), which instead of outputting 0 for negative input values uses a learnable parameter a to create a linear slope. This parameter is usually limited to small values to maintain the properties of the ReLU activation function. The PReLU is defined by equation 2.5.

$$\sigma(x_i) = \begin{cases} a_i x_i & : x_i < 0 \\ x_i & : x \geq 0 \end{cases} \tag{2.5}$$

Softmax

The softmax function is used to scale the outputs y_l of the final layer between 0 and 1, where the sum of all outputs is 1. This can be interpreted as a pseudo probability. The softmax function is defined by equation 2.6.

$$softmax(\mathbf{y}_l) = \frac{exp(\mathbf{y}_l)}{\sum_{i=1}^{N_l} exp(y_l^i)} \tag{2.6}$$

2.1.4 Loss function

To evaluate how well the network predicts the data a loss function is used to calculate the difference between the final predictions and the provided ground truth (14).

Training

The network is trained iteratively on batches of input. For each iteration the network is evaluated on m inputs $\mathbf{x}^{(1)}, ..., \mathbf{x}^{(m)}$ and corresponding ground truth $\mathbf{z}^{(1)}, ..., \mathbf{z}^{(m)}$. An loss metric is calculated between the predicted and expected outputs. The weights are then updated to minimize the loss. Different optimizers can be used to minimize the loss. One of these is stochastic gradient descent (14).

Let $f\left(\mathbf{x}^{(i)}; \boldsymbol{\theta}\right)$ be the output of the network with parameters $\boldsymbol{\theta}$ (weights, biases and other learnable parameters) and $\mathbf{x}^{(i)}$ as input. Let $L\left(f\left(\mathbf{x}^{(i)}; \boldsymbol{\theta}\right), \mathbf{z}^{(i)}\right)$ be some loss function, such as cross entropy or a quadratic loss function. The gradient estimate of the parameters $\boldsymbol{\theta}$ is then calculated by equation 2.7.

$$\hat{\boldsymbol{g}}_k = \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m} L\left(f\left(\mathbf{x}^{(i)}; \boldsymbol{\theta}\right), \mathbf{z}^{(i)}\right) \tag{2.7}$$

The parameters are then updated by a learning rate ϵ_k multiplied with the gradients added to the previous parameters, $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_{k-1} - \epsilon_k \hat{\boldsymbol{g}}_k$, where k is the iteration number. The gradient of the parameters are obtained using backpropagation (14), which uses the chain rule to propagate the loss from the final layer backwards through all layers.

The learning rate ϵ_k determines how fast the network learns. Using a constant learning rate during the whole training can lead to slow convergence or lack of convergence. Learning rate schedules are used to alter the learning rate during training based on number of epochs and/or training scores. This makes it possible to use a relatively larger initial learning rate for faster convergence and lowering the learning rate towards the end of the training to get more fine-grained updates of the parameters. A momentum can be used to update the gradients based on a form of average of gradients computed for a number of consecutive optimization steps, instead of applying the gradients directly each optimization step. This has shown to make the training more robust against local minimas (49).

Convolutional neural networks

Convolutional neural networks (CNNs) introduce the convolutional layer and are similar to the fully connected neural networks previously presented. They both consists of neurons, learnable weights, use loss function and can be trained using the same concepts. They differ in how the weights and outputs are constructed, where convolutional layers are able to exploit the spatial information found in images. A typical CNN is made up of several sequential layers connected by weights acting as convolutional filter kernels. Each layer $l \in [1, L]$ contains C_l feature maps (FMs) (21). In 2D CNNs each FM is a 2D grid of neurons that is used to detect features in the FM of the previous layer.

Let $\mathbf{k}_l^{m,n}$, called a kernel, be a matrix of learned weights $\mathbf{W}_l^{m,n}$ from the the n-th FM in the previous layer to the m-th FM in layer l. The activations of the m-th FM in the l-th layer can then be represented by equation 2.8 (21), where $\star$ is the convolution operator.

$$\mathbf{y}_l^m = \sigma\left(\sum_{n=1}^{C_{l-1}} \mathbf{k}_l^{m,n} \star \mathbf{y}_{l-1}^n + b_l^m\right) \tag{2.8}$$

This can be seen as convolving each FM of the previous layer by a 2D kernel filter and summing the result. Figure 2.7 shows this operation. To obtain all the activations the kernels are slided across the FMs, meaning that all neurons in the m-th FM of the l-th layer will share the same weights $\mathbf{W}_l^{m,n}$, $n \in [1, C_{l-1}]$. All FMs in a layer has the same dimensions and all kernels $\mathbf{k}_l^{m,n}$,$m \in [1, C_l]$,$n \in [1, C_{l-1}]$ have equal size. Let K_l be the number of weights of a kernel in the l-th layer. A 2D 3x3 kernel will require $K = 3^2 = 9$ weights. The total amount of weights between two layers is then given by $C_{l-1}C_lK$.

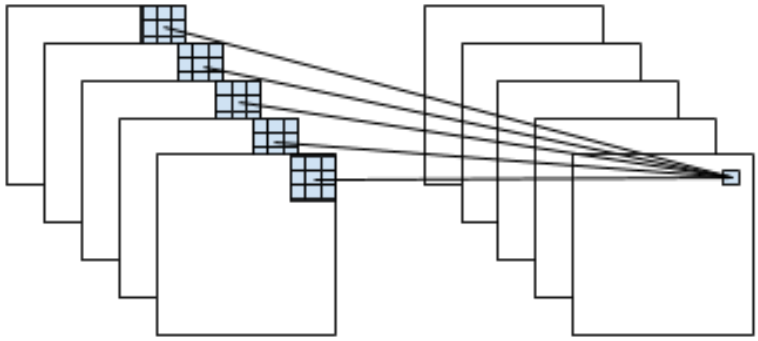


Figure 2.7: 2D Convolutional layer with a 3x3 kernel size and 5 FMs in each layer. Each line represents 9 weights but has been simplified for the illustration.

This shows the concept of 2D convolutional layers, but this can be easily extended to 3D. The 2D kernels are changed to 3D kernels and the 2D FMs are replaced by 3D FMs. This is illustrated by figure 2.8.

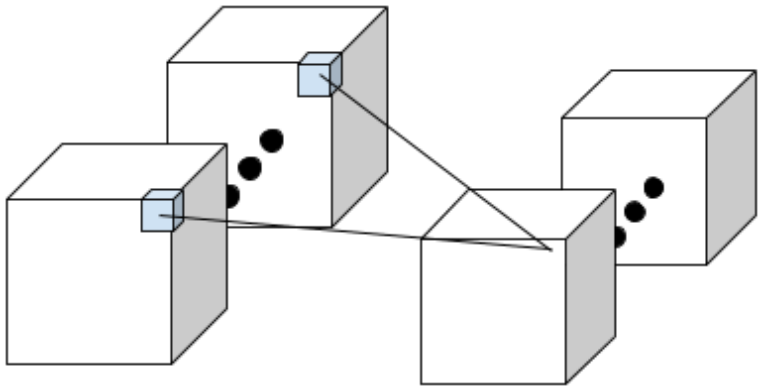


Figure 2.8: 3D Convolutional layer with 3D kernels. Each line represents a K weights. The dots represents multiple 3D FMs.

2.2 Literature Review

In this section relevant literature to the project is presented. Literature on coronary artery segmentation is presented first, divided into methods for general vessel segmentation, coronary artery centerline extraction and coronary artery lumen segmentation. This is followed by relevant literature on existing deep learning approaches for coronary artery segmentation as well as brain tumor segmentation.

2.2.1 Related work for coronary artery segmentation

Segmentation of the coronary arteries is an important step for improving visualization and quantification of the vessels. There has already been done a substantial amount of research on the segmentation of the coronary arteries as a result of this. Lesage et al.(25) provides an extensive review of algorithms for 3D vessel segmentation in general. A review of algorithms specifically for the problem of coronary artery segmentation is given by Dehkordi et al. (7). Most of these methods have been evaluated on the Rotterdam Coronary Artery Algorithm Evaluation Framework for Centerline Extraction (41). In this framework the algorithms are evaluated on three challenges: automatic, minimal user interaction and interactive extraction. A score is assigned to each method based on an overlap measure and an accuracy measure. The scores of the methods are published and ranked against each other, making it possible to perform quantitative comparisons of the algorithms.

The highest ranked method of all the challenges is presented by Friman et al. (11). This method uses a combination of multiple hypothesis tracking of vessels, minimal paths and user interaction in the form of adding points to guide the algorithm. By considering several hypothetical paths and bridging gaps based on minimal paths, the method is able to extract difficult sections. To obtain good results with the method, a user is required to manually add points to guide the extraction. This requires a user with knowledge about the coronary arteries and the task of manually adding points can be time consuming. Several fully automatic methods for centerline extraction has been proposed in order to avoid these issues.

Fully Automatic Centerline Extraction

The fully automatic methods can be divided into two categories, model-driven and data-driven. A tracking-based method is proposed by Tek et al. (47). The first step of this method is to detect the aorta, which is then used as a mask for finding ostia (the positions on the aorta where the coronary arteries originates from). The centerline is then extracted by tracking with a multi-scale medialness algorithm to the end of all branches. To perform the actual centerline extraction, a graph based optimization method is used. Kitslaar et al. (24) proposed a method of first segmenting the heart and aorta and using connected component analysis to find candidate coronary regions. Starting from the aorta, a variant of region growing is used to segment the complete tree. Zambal et al. (54) proposed a method using a model-driven method to first segment the complete heart. The model is then used to generate candidate positions for the coronary arteries. Depth first search is then used together with small cylinder shaped models to extract the complete coronary tree.

A data-driven approach is proposed by Yang et al. (52). This method uses an improved vesselness filter based on Frangi et al.(10) by using ray-casting to mitigate issues at bifurcations where the arteries branch. The vesselness filter generates responses across the entire volume, giving stronger responses for structures that have a tubular structure. The responses are thresholded and candidate centerlines are obtained through skeletonization.

Hough circle transform is used to segment the aorta partially, which is then used to find the starting points of the centerlines. Centerlines are then grown from the ostium using branch searching. In the last step, the contour of the lumen walls in multi-planar reconstructions are detected and used to refine the centerline.

Bauer et al.(4) proposes a data-driven method based on eigenvalue analysis of the Hessian matrix and coronary tree reconstruction. It performs a tube detection filter to detect tubular structures in the image and use a ridge traversal algorithm to obtain candidate centerlines. The tube detection filter used is based on Frangi et al.(10). In order to use this filter to detect tubular structures with a larger diameter, the method uses gradient vector flow(51). As the tube detection filter used in this method assumes that vessels have a circular cross section, the filter will fail to detect junctions and heavily calcified parts. The method introduces a tree reconstruction algorithm which tries to create a minimum spanning tree of the candidate centerlines based on a cost function. The coronary arteries are then selected as the longest connected structure.

Zheng et al.(56) is based on combining model-driven and data-driven approaches. The method uses exploits anatomical knowledge about the coronary arteries by utilizing segmented heart chambers to detect initial positions for centerlines and constrain them. A pre-learned shape model is then used to refine the centerlines. To extend the centerlines a data-driven method is used. Compared to other methods evaluated using the centerline extraction evaluation framework(41), it outperforms all other methods in terms of accuracy measure. For the overlap measure it ranks second among the fully automatic methods.

All the fully automatic methods presented in this section, with the exception of the method proposed by Bauer et al. (4), includes a step for segmenting the aorta and/or heart to initialize or restrict the centerline extraction.

Coronary lumen segmentation algorithms

Segmentation of the coronary arteries is an important step for improving visualization and quantification of the vessels. As a result, a large amount of algorithms for coronary artery segmentation has already been proposed. Lesage et al.(25) provides a review of 3D vessel segmentation algorithms. For this project we will look at fully automatic segmentation methods that has been evaluated using the previously mentioned evaluation framework(22; 41). The top ranking fully automatic lumen segmentation methods can be separated into two categories, graph-cut based and level-set based.

One of the methods that is evaluated using the framework from the second challenge is Lugauer et al.(30). In this method an algorithm for lumen segmentation based on Markov Random Field and learning-based boundary detection is presented. A Min-cut algorithm is then used to obtain the optimal surface generation. This method is fully automatic and is shown to perform better than the human observers. A similar method is proposed in Lugauer et al.(29), which is also uses a learning-based boundary detection using cascades. In this method the calcifications are explicitly removed in the boundary detection step. The method then uses a surface optimization scheme as proposed by Li et al.(26). Kitamura et

al.(23) uses a method based on multi-label graph cuts. Higher order potentials are used as shape priors. Candidate shapes are detected using Hessian analysis and using knowledge about healthy lumen being mostly tubular.

Mohr et al.(34) uses an unsupervised classification technique based on Bayesian Information Criterion to segment calcifications. Then the tissue is classified with probabilities of being part of the lumen or vessel wall. For the segmentation of the lumen, a level-set approach is used based on the result of the tissue and calcification classifications. Wang et al.(50) proposes a method using an implicit 3D model of an already extracted centerline. A level-set method is then used to propagate the growth of the model. New centerlines are computed and the diameter of the vessels are re-estimated. This is done iteratively until convergence.

Antiga (2) presents a framework for modelling vessels and blood flow based on medical images of larger arteries. A semi-automatic method based on level-sets is used to segment the vessels. Marching cubes is used on the segmentation result to create a surface model of the vessel. The models are then further processed, emphasizing the importance of having good models before they can be converted to meshes used for computational fluid dynamics.

2.2.2 Deep Learning for medical image segmentation

In the last years there has been a large increase in papers on deep learning for medical imaging. Litjens et al. presents a survey paper reviewing over recent 300 papers, of which 240 were published in 2016 and 2017. The survey paper presents different types of deep learning, such as deep belief networks (DBN), stacked auto encoders (SAE) and convolutional neural networks (CNN). The papers are then categorized based on different types of tasks. These tasks include classification, where a network takes one or more images as input and outputs a small amount of variables used for diagnostics. A variable can as an example be as simple as outputting if a disease is present or not. Another task is detection, where a network takes images as input and the output is used to localize organs and landmarks. The result can be positions, bounding boxes and other variables describing the location and orientation of anatomical structures in the images. For the task of performing registration of multiple images, neural networks has been used both to output a similarity metric between the two images, as well outputting the direct transformation that need to be performed on the images. Another task is segmentation, where a network takes images as input and classifies each pixel/voxel as output.

The methods proposed for segmentation are further categorized based on their approaches. One of the approaches is to use a small patch of the image as input to the network. To segment the whole image, patches are extracted in a sliding-window manner and evaluated by the network. this does however lead to a large amount of extra computations where the image patches overlap due to valid convolutions. Long et al. (28) proposes a fully convolutional neural network (fCNN) by representing fully connected layers as convolutions. This makes it possible to use larger images as input than the network was trained on. The output is however smaller than the input image due to downsampling in the form

of max-pool layers. To fix this Long et al. (28) proposes a method of shifting the image one pixel in each dimension and performing segmentation. This is repeated the same number of times as the scaling factor and the outputs are then interlaced together to obtain the full resolution result. Ronneberger et al. (39) presents a network named U-net, which solves the problem of resolution by using an equal amount of upsampling layers as downsampling layers. The architecture consists of a contractive part and an expansive part. The network uses skip-connections between before and after the downsampling and upsampling to the same level. The architecture is extended to 3D by Cicek et al. (6) by replacing the 2D convolutional layers with 3D convolutional layers. Milletari et al. (32) proposes a similar 3D architecture based on U-net and introduces a Dice-loss layer. The downsampling and upsampling allows these architectures to work on the full image. This means that large intermediate featuremaps needs to be stored at each layer. For 3D images the memory requirements increases drastically and the methods proposed by Cicek et al. (6) and Milletari et al. (32) describes using 1 and 2 volumes respectively each training batch. To compensate for this, a large momentum is used while training. For both methods, the training data is augmented on-the-fly by performing elastic deformation on both the input and the ground truth.

To avoid the large memory requirements that are present in the methods previously mentioned, other methods have used a multi-stream architecture. These are used mainly for two purposes. The first is to perform 2.5D segmentation. Instead of using 3D convolutional layers on the full input volume, Setio et al. (42) uses multiple distinct streams of 2D convolutional layers operating on differently angled slices. The streams are then merged towards the end of the network. The other purpose of multiple streams are to provide segmentation on multiple scales. Kamnitsas et al. (21) proposes a method using two streams which they call a dual pathway. One stream operates on local information from a small patch, while the other stream works on a downscaled version of a larger section around the small patch. Having two distinct streams working on different scales makes it possible for the network to detailed local information together with information about larger surrounding structures without severely increasing the number of parameters or the size of the featuremaps. This also makes it possible to train the network on smaller patches and use larger patches when running, which in turn allows the network to have a larger batch size compared to the methods working on the full image (28; 39; 6; 32). Since these methods are training on the whole image class balance becomes an issue. To compensate for this a weighted loss function together with a weight map is introduced (39). Since the dual pathway based methods (21) can work on smaller patches, a non-uniform sampling scheme is used to alleviate issues related to class imbalance.

Litjens et al. (27) discusses a challenge that is common to the segmentation methods mentioned, which is that the methods will often create spurios responses in the output. Some methods use a post-processing step on the prediction maps that the networks generate. Shakeri et al. (43) presents a method that uses a 2D fCNN to to segment structures in MR scans of the brain. The segmentation is further improved by using Markov Random Field on the prediction map output of the CNN. Kamnitsas et al. (21) uses a similar method of improving the segmentation, but with a 3D Conditional Random Field instead.

Deep learning methods for coronary artery segmentation

When it comes to using deep learning for coronary artery segmentation there have been several proposed methods. Gulsun et al. (15) presents a fully automatic method for extracting the centerlines of the coronary arteries. This method uses a computed flow field to find optimal centerline paths. After several centerline proposals has been made, a multi-channel 1D CNN is trained and used to classify centerlines leaking into other non-coronary vessels so that they can be pruned. The input consists of multiple profiles sampled along the centerline, where each channel is a profile such as image intensity, vesselness, curvature and gradient statistics among others. Moeskops et al. (33) presents a method of using a multi-stream CNN working on inputs of three orthogonal 2D slices. In this paper the CNNs ability to automatically learn features from one type of images, such as brain MRI images, and applying these when classifying images of a different type, in this work breast MRI and cardiac CTA. This is done by training the CNN on multiple types of images at the same time. For the cardiac CTA dataset the task is segmentation of the coronary arteries.

Deep learning methods for brain tumor segmentation

Litjens et al. (27) mentions several deep learning methods used for brain tumor segmentation. Havaei et al. (16) proposes a method of using a multistream 2D CNN, where each stream uses a different MR modality as input. The streams are merged together in an abstraction layer and further processing is done by subsequent layers. The network is trained to handle missing input for some of the modalities at inference time. Pereira et al. (37) used a 2D CNN with different MR modalities represented as separate channels in the input. Zhao et al. (55) presents a multistream 2D CNN with three pathways. The input of the pathways are 2D patches at three separate scales. Another method for brain tumor segmentation is presented by Kamnitsas et al. (21), using 3D convolutions on patches of two separate scales in a mutlistream CNN and was the winner of the ISLES 2015 challenge (31).

Chapter 3

Method

In this chapter the chosen method is presented. The chosen deep learning method is explained, followed by a summary of the different datasets that are used in this thesis. A section is used to describe the preprocessing steps applied to the datasets. The different experiments that were set up are then explained, first experiments for coronary artery segmentation, then brain tumor segmentation and segmentation on the digital rocks dataset. Finally the metrics used to evaluate the methods are presented.

3.1 Deep learning framework

Based on the literature review the deep learning framework presented by Kamnitsas et al. (21) was chosen. The original paper introduces a network architecture named DeepMedic, which is a 3D CNN using a dual pathway of two different scales. The DeepMedic configuration has since been updated by the original authors based on new discoveries of architecture choices and training hyperparameters that work well on brain tumor segmentation. The method proposed in the original paper (21) uses a conditional random field (CRF) to refine the output of the network. In order to give a stronger emphasis on the network learning the correct segmentations, the CRF was not used.

3.1.1 Architecture

The full architecture of the DeepMedic network is shown in figure 3.1. The network consists of two streams with separately learned weights. One of the streams works on the normal resolution while the other stream downscales the input 3 times before performing convolutions. Outside of this, both pathways are identical in terms of the number of convolutional layers and FMs per layer. The network uses residual connections as defined by He et al. (18). The input for the 3rd convolutional layer is added to the outputs of the 4th layer. These are added together after the convolution of the 4th layer, but before applying the non-linearity σ. Residual connections are also used for the 6th and 8th layers. These

are represented by connections with a $\oplus$ symbol. Each convolutional layer in both pathways performs only valid convolutions, meaning that the dimensions are reduced by 2 for each 3^3 convolution. After 8 convolutional layers the FMs of the two pathways are concatenated and a 3^3 kernel with padding is used to maintain the size of the FM. Two fully connected layers with 1^3 convolutions are then used, where the last layer has a number of FMs equal to the number of output classes the network should be able to predict. The DeepMedic architecture and software framework are used for all networks that are trained in this thesis.

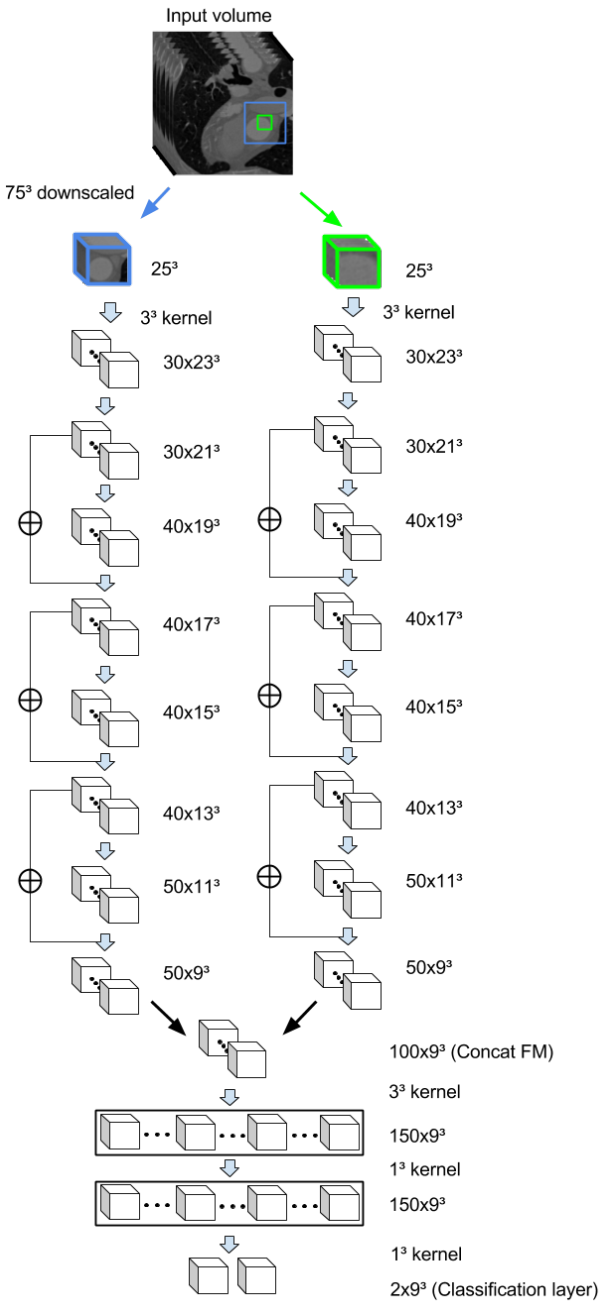


Figure 3.1: Overview of the architecture used. The architecture uses two identical pathways working on different scales of input. The architecture is 11 layers deep and uses residual connections, represented by connections with $\oplus$. The FM descriptions are on the form $(number\ of\ FMs)$x$(size\ of\ FM)^3$. Networks classifying 3 classes will have 3 FMs in the classification layer.

3.1.2 Training hyperparameters

Since the network architecture does not use any downsampling or upsampling layers, the size of each voxels receptive field is small. This means that only a small patch equal to the size of the receptive field of each voxel is necessary to train the network. Using 1^3 convolutions for the fully connected layer makes it possible to train on multiple voxels each inference step. Kamnisas et al. (21) uses this to provide a hybrid training scheme, where the network is trained on small image patches, but is able to perform inference on larger image segments. It is however limited by memory as the size of the FMs will increase with increased input size. The weights are not effected by this.

The network samples patches for training randomly from the entire volume. To alleviate issues with class imbalance the network selects the patches with an equal probability of the central predicted voxel being centered on a specific class.

The network is trained for 35 epochs, where each epoch is made up of 20 subepochs. For each subepoch, 1000 training patches are extracted equally from the provided training volumes. A batch size of 10 is used. After all samples in the training batch has been processed, one optimization step is performed.

Dropout is used on the last two fully connected convolutional layers. For both layers, each weight has a 50% chance of being temporarily removed each optimization step.

The weights are initialized by sampling from the normal distribution $\mathcal{N}(0, \sqrt{2/n_l^{in}})$, where n_l^{in} is the number of weights a neuron on layer l is connected to from the input of the layer, given by equation 3.1. This is done to better preserve the signals in the first training steps (17).

$$n_l^{in} = C_{l-1} \prod_{i=\{x,y,z\}} \mathbf{k}_l^{(i)} \tag{3.1}$$

Batch normalization is used for each hidden layer to relieve the effects of the internal covariate shift (19). The batches are normalized after each hidden layer based on the average and standard deviation of all patches in the batch, but when performing inference the network must be able to handle a single patch. A rolling average over 60 batches are used to obtain the average and standard deviation used for inference.

Training is performed with the RMSProp optimizer (49) with nesterov momentum (46). The RMSProp parameters that are used are $\rho_{RMS} = 0.9$ and $\epsilon_{RMS} = 1e-4$. A momentum value of 0.6 is used for the nesterov momentum.

L1 regularization is used with a value of 0.000001. L2 regularization is used with a value of 0.0001.

The initial learning rate is 0.001 and a learning rate schedule based on dividing the learning rate by 2 at predefined epochs is used. The predefined epochs are epochs: 12, 16, 19,

22, 25, 28, 31 and 34.

3.2 Datasets

The main focus of this thesis is to explore the use of deep learning on automatic segmentation of the coronary arteries. For this task, three different datasets has been used for both training and evaluating the networks. The first dataset is provided by the Rotterdam Coronary Artery Algorithm Evaluation Framework for centerline extraction (41). It consists of 32 CT-volumes of which 8 is used for training and 24 is used for testing. A list of points describing the position and radius of the centerlines for four selected vessels are provided for each of the training volumes.

The second dataset is provided by the Rotterdam Coronary Stenoses Detection and Quantification Evaluation Framework (22). It consists of 48 CT-volumes, 18 for training and 30 for testing. Ground truth data for stenoses position and quantification are provided for the training volumes. Positions of the centerlines for all volumes are provided but lacks information about the radius of the vessels. In addition to this, 3D models for different parts of the heart, including the aorta, are provided for visualization.

The third dataset that has been used in this thesis is provided by the Coronary FFR project pilot project. It consists of 8 CT-volumes. A 3D model of the lumen surface for a part of the coronary tree has been manually segmented by a clinical expert for 3 of the CT-volumes.

The DeepMedic architecture has been verified by training and testing it on a dataset provided by USIGT (12). It consists of 20 MR-volumes with manual segmentations of glioblastoma performed by two clinical experts.

The rock dataset is provided bu FEI (1). It consists of 3D microscopy volumes of 3 different types of rock. For each dataset a labelled volume has been created by a semi-automatic method.

The distinction between training and testing data is made clear in the two first datasets provided by the evaluation frameworks. For the three other datasets the distinction has to be done manually to ensure that the network has enough data to train on and at the same time is validated against a diverse set of testing data that it has not been trained on to ensure that the network is able to handle new data. A summary of the datasets is shown in table 3.1.

3.3 Data Preprocessing

The medical datasets have been acquired by different machines and settings. This leads to a large variation in both intensity values and resolution. The CT volumes are made up of axial slices, where each slice has a resolution of 512x512 pixels. The number of slices

Dataset	# Training	# Testing	Modality	Dimensions	Ground truth
Centerlines	8	24	CT	512x512xDepth	Centerline position and radius
Stenoses	18	30	CT	512x512xDepth	Stenoses position and quantification. Centerline position. 3D model of aorta.
Pilot Project	0	3	CT	512x512xDepth	3D model of lumen surface
Glioblastoma	13	4	MR	200x200xDepth	Labelled ground truth volume
Rock	-	-	CT	1000x1000x2499	Labelled ground truth volume

Table 3.1: Summary of datasets used in this thesis

differs for each volume, ranging from 260 slices to 600 slices. The volumes also have different pixel spacing and distance between slices. Although smaller pixel spacing leads to greater detail in the image, it can cause trouble when training the neural networks. The neural networks do not use information about pixel spacing in any way. If we have two volumes where the second volume has half the slice spacing but twice the number of slices as the first, the structures in the second volume will be twice as long when seen by the neural network. The solution to this is to resample all the volumes to have the same pixel spacing and spacing between slices, as recommended by Kamnitsas et al. (21).

3.3.1 Resampling

Let **s** be the scaling factor defined by equation 3.2, where $spacing\ x$, $spacing\ y$, $spacing\ z$ is the spacing in mm between the voxels in the x, y and z directions respectively.

$$\mathbf{s} = \begin{pmatrix} spacing\ x_{original} / spacing\ x_{target} \\ spacing\ y_{original} / spacing\ y_{target} \\ spacing\ z_{original} / spacing\ z_{target} \end{pmatrix} \tag{3.2}$$

Let **d** be a vector that represents the size of a volume in number of voxels as described by equation 3.3.

$$\mathbf{d} = \begin{pmatrix} width \\ height \\ depth \end{pmatrix} \tag{3.3}$$

The target size is defined by equation 3.4, where $\odot$ represents a component-wise multiplication between the original size and the scaling factor. The elements are rounded down to the nearest integer to ensure the number of voxels to be discrete.

$$\mathbf{d}_{target} = \lfloor \mathbf{d}_{original} \odot \mathbf{s} \rfloor \tag{3.4}$$

Let $I_{resampled}$ be the resampled image with size defined by $\mathbf{d}_{target}$. Let **v** be an arbitrary position in the resampled image. We can represent the normalized position $\mathbf{v}_t$ as shown in equation 3.5. The corresponding position in the original volume is given by $\mathbf{v}_s$ as shown

in equation 3.6.

$$\mathbf{v}_t = \begin{pmatrix} x_t \\ y_t \\ z_t \end{pmatrix} = \mathbf{v} \odot \frac{1}{\mathbf{d}_{target}} \tag{3.5}$$

$$\mathbf{v}_s = \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix} = \mathbf{v}_t \odot \mathbf{d}_{original} \tag{3.6}$$

The value for a voxel in the resampled image is then performed with trilinear interpolation, as shown by equations 3.7, 3.8, 3.9 and 3.10.

$$\begin{aligned}
C_{000} &= I_{original}((\lfloor x_s \rfloor, \lfloor y_s \rfloor, \lfloor z_s \rfloor)^T) \\
C_{001} &= I_{original}((\lceil x_s \rceil, \lfloor y_s \rfloor, \lfloor z_s \rfloor)^T) \\
C_{010} &= I_{original}((\lfloor x_s \rfloor, \lceil y_s \rceil, \lfloor z_s \rfloor)^T) \\
C_{011} &= I_{original}((\lceil x_s \rceil, \lceil y_s \rceil, \lfloor z_s \rfloor)^T) \\
C_{100} &= I_{original}((\lfloor x_s \rfloor, \lfloor y_s \rfloor, \lceil z_s \rceil)^T) \\
C_{101} &= I_{original}((\lceil x_s \rceil, \lfloor y_s \rfloor, \lceil z_s \rceil)^T) \\
C_{110} &= I_{original}((\lfloor x_s \rfloor, \lceil y_s \rceil, \lceil z_s \rceil)^T) \\
C_{111} &= I_{original}((\lceil x_s \rceil, \lceil y_s \rceil, \lceil z_s \rceil)^T)
\end{aligned} \tag{3.7}$$

$$\begin{aligned}
C_{00} &= (1 - x_t)C_{000} + x_t C_{001} \\
C_{01} &= (1 - x_t)C_{010} + x_t C_{011} \\
C_{10} &= (1 - x_t)C_{100} + x_t C_{101} \\
C_{11} &= (1 - x_t)C_{110} + x_t C_{111}
\end{aligned} \tag{3.8}$$

$$\begin{aligned}
C_0 &= (1 - y_t)C_{00} + y_t C_{01} \\
C_1 &= (1 - y_t)C_{10} + y_t C_{11}
\end{aligned} \tag{3.9}$$

$$I_{resampled}(\mathbf{v}) = (1 - z_t)C_0 + z_t C_1 \tag{3.10}$$

The ground truth labeled volume is similarly resampled, except the trilinear interpolation step is replaced by nearest neighbor as shown in equation 3.11.

$$G_{resampled}(\mathbf{v}) = G_{original}((round(x_s), round(y_s), round(z_s))^T) \tag{3.11}$$

3.3.2 Normalization

After resampling, the volume is normalized to have intensity values between 0 and 1. This is done by dividing all intensity values in the volume by the highest intensity value as shown in equation 3.12.

$$I_{norm}(\mathbf{v}) = \frac{I_{resampled}(\mathbf{v})}{\max_{\mathbf{v}} I_{resampled}(\mathbf{v})} \tag{3.12}$$

The intensities are then centered around zero by subtracting the mean value and divided

by the standard deviation as suggested by Kamnitsas et al. (21). Equation 3.13 gives the final prerpocessed volume, where $mean$ gives the mean intensity of I_{norm} and std give the standard deviation.

$$I_{preprocessed} = \frac{I_{norm}(\mathbf{v}) - mean(I_{norm}(\mathbf{v}))}{std(I_{norm}(\mathbf{v}))} \tag{3.13}$$

3.4 Coronary artery segmentation

To train the neural network to segment the coronary arteries we need to provide training data. The stenoses dataset provides some ground truth about the lumen surface in the form of models of cross sections at 4 locations in each of the training datasets. It is however hard to convert these cross-sectional planes into a labeled volume, which is required by the neural network. The dataset also provides the positions of the centerlines, but lacks any information about the radius. The centerline dataset does however provide both the centerline position and radius and was chosen as training data.

3.4.1 Generating training data

To use the use the centerline data to train the neural network it first needs to be converted to a labeled volume. For each volume in the training set, a new labeled volume with the same dimensions as the training volume is created and initialized to only have background labels. Each point on the centerline contains a position and a radius. For each point a sphere with the corresponding radius is drawn in the labeled volume and marked as part of the coronary arteries. The end result is an approximation of the lumen, assuming a close to uniform tubular shape. This is illustrated by figure 3.2.

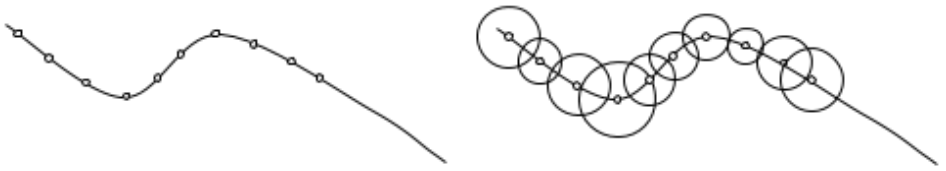


Figure 3.2: Simplified illustration of how a ground truth volume can be generated from overlapping spheres based on position and radius of centerline. The distance between the spheres are for illustration purposes as they are much closer in the real centerline data.

3.4.2 Training the network

After generating labels for the 8 training volumes, the volumes are preprocessed by resampling and normalization as described in Section 3.3. Two different settings of voxel spacing has been chosen for resampling. The first is 0.40mm in all dimensions, which is close to the original images. This is the baseline volume. For the second setting the voxel spacing has been doubled to 0.80mm, essentially downscaling the volumes by a factor of

2. This is done to test the effects of different resolutions on the neural network. It is also a way to increase the receptive field of the neural network without introducing more weights.

While training the neural network it is preferrable to perform validation while training to observe how well the network is able to generalize. Since there are only 8 training volumes with ground truth labels, 7 are used to train the network while the remaining volume is used for validation.

Due to the limited amount of volumes with ground truth, an experiment with data augmentation during training was performed. The data augmentation consists of randomly mirroring the input patches around all three axises. This essentially increases the available data by 8 times.

The coronary arteries makes up only a small part of the entire CT volume, which means that the patches extracted from the background will have a larger variation than the patches centered on the coronary arteries. There are also other side branches of the coronary arteries that are not labelled in the ground truth. An experiment was performed by providing a region of interest (ROI) around the coronary arteries in the ground truth. The same data augmentation as the previous experiment is used due to smaller amount of possible samples caused by the ROI. Figure 3.3 illustrates the concept. The training patches are only sampled from within the ROI of 12 voxel radius around the labelled arteries.

The networks trained in the experiments are named $CoronaryCNN_{(experiment)}$ and are summarized by table 3.2.

Experiment	**Description**
$CoronaryCNN$	Baseline
$CoronaryCNN_{0.80mm}$	Resampled to 0.80mm voxel spacing
$CoronaryCNN_{flip}$	Data augmentation by randomly mirroring (flipping) in all directions each sample
$CoronaryCNN_{ROI}$	ROI of 12 voxels, data augmentation like $CoronaryCNN_{flip}$

Table 3.2: The experiments used for training the CoronaryCNN

3.4.3 Extracting the coronary arteries

Ideally the neural network should be able to take an input volume and output a labelled volume of the vessels. However, the networks has a tendency to produce a large amount of responses that are not part of the coronary arteries. An assumption is made that the networks segmenting the coronary arteries manages to segment the vessels as connected structures. This means that if the starting point of each part of the coronary tree is given, it is possible to use information about connected components to extract only the coronary arteries.

Figure 3.3: Region of interest around labelled coronary arteries.

In this thesis, a new method for refining the segmentation output to only include the coronary arteries is proposed. The proposed method use two separately trained networks. One network is trained on aorta segmentation and the other is trained on coronary artery segmentation. After obtaining a segmentation from the network trained on coronary arteries, the aorta segmentation is used to find the intersection between the aorta and the coronary arteries. This intersection is then used as start positions to select the coronary arteries and discarding spurious responses. The process is illustrated by figure 3.4, where AortaCNN and CoronaryCNN are trained networks for aorta and coronary artery segmentation respectively. The postprocessing of the aorta consists of finding the largest connected component and discarding the other responses.

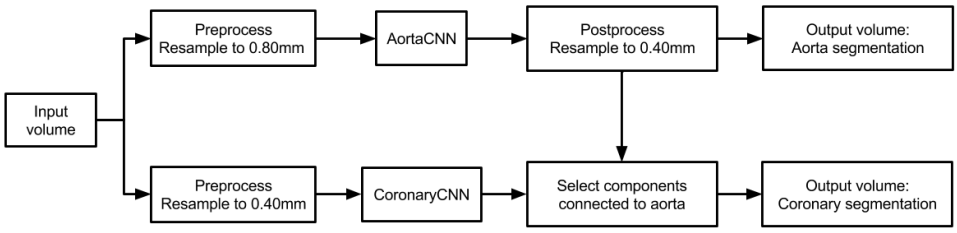


Figure 3.4: Method for extracting the coronary arteries and removing spurious responses in the CNN output segmentation.

3.4.4 Aorta segmentation

The dataused to train a network to segment the aorta is provided by the Rotterdam Evaluation framework for stenoses detection. The data is provided in the form of 3D models, which is then sampled to a label volume using functionality in 3D Slicer (38). The images and ground truth are then preprocessed. The target spacing to resample the volumes to has been chosen to be the same as the spacing used for the coronary artery extraction network, which are 0.40mm and 0.80mm. The experiments are summarized in table 3.3.

Experiment	Description
$AortaCNN_{0.40mm}$	Volumes are resampled to 0.40mm voxel spacing.
$AortaCNN_{0.80mm}$	Volumes are resampled to 0.80mm voxel spacing.

Table 3.3: Summary of experiments for aorta segmentation

3.4.5 Testing on Pilot Project dataset

The pilot project at St. Olavs Hospital provides 8 CT volumes, of which 3 has been manually labelled by a clinical expert. The manual segmentation consists of one model containing the aorta and the coronary artery tree for each volume. The models are converted to binary volumes using 3D Slicer (38). The CT volumes are resampled to 0.40mm and 0.80mm and preprocessed. To evaluate on the pilot dataset, the outputs of the proposed method for extracting the coronary arteries (aorta and coronary segmentation) are combined into one binary volume.

3.5 Brain tumor segmentation

The brain tumor dataset consists of manual segmentations performed twice by two clinical experts using three different programs. A network based on the DeepMedic architecture is trained and validated using the second manual segmentations by A.L.S using slicer (12). The manual segmentations were performed on the T1 weighted MRI volumes. The dataset also contains FLAIR volumes, but these were not used for training as not all of them were co-registered to the T1 weighted volumes. Due to missing manual segmentation for three of the MRI volumes, the network was trained on 13 volumes and validated on 4. All MRI

volumes are preprocessed and resampled to 1mm voxel spacing. The network trained for brain tumor segmentation is referred to $BrainTumorCNN$ in this thesis.

3.6 Digital rock segmentation

The rock dataset consists of three volumes of different types of rock. All three volumes are too large to fit in memory and has been divided into non-overlapping subvolumes. Each subvolume has a size of 175x175x175 voxels. For each of the three volumes 150 subvolumes are extracted, of which 40 are used for training and 10 are used for validation. The three types of rock are:

- **Type 1**: Bentheimer sandstone (BT)
- **Type 2**: Berea sandstone (BR)
- **Type 3**: Carbonate (CA)

In the first experiment the baseline network is trained on different amounts of training data of bentheimer sandstone. By doing this it is possible to observe the effect of limited amount of data compared to a larger amount. The networks trained in the different experiments are described in table 3.4.

Experiment	# Training subvolumes	# Testing subvolumes
$RockCNN_1$	1	10
$RockCNN_{10}$	10	10
$RockCNN_{40}$	40	10

Table 3.4: Experiments with different number of training subvolumes of the Bentheimer volume.

The second set of experiments are used to observe how well a network trained on one type performs on a type it has not seen before. $RockCNN_{BT}$ (which is the same network as $RockCNN_{40}$) is first tested on the other two datasets. Then a second network is trained on two types and tested on all three. Lastly a network is both trained and tested on all three types. Table 3.5 shows the details of the three experiments.

	# Training			# Testing		
Experiment	**BT**	**BR**	**CA**	**BT**	**BR**	**CA**
$RockCNN_{BT}$	40	0	0	10	10	10
$RockCNN_{BT+BR}$	40	40	0	10	10	10
$RockCNN_{BT+BR+CA}$	40	40	40	10	10	10

Table 3.5: Experiments with training on different types of rock.

3.7 Implementation details

Kamnitsas et al. (21) provides the source code of the deepmedic framework, which is used for everything related to neural networks for the methods in this thesis. The framework

is programmed in Python and uses Theano (48) as neural network library. Configuration files are used to modify the hyperparameters of the network and alter the settings used for training. Tools for plotting the training and validation metrics are provided by the framework.

The ground truth data generation from centerlines was programmed in FAST (44) using C++ and OpenCL. 3D Slicer (38) was used to convert between 3D models and binary labelled volumes. All other methods described in this chapter were programmed using Matlab.

The training and evaluation of the networks were performed on machines with NVIDIA GTX 1080 graphical processing units (GPUs).

3.8 Evaluation metrics

To evaluate how well the methods perform segmentation there are several metrics that can be used. For each input volume the network predicts a class for each voxel. This is then compared against the ground truth, which is a volume of the same size as the input volume where each voxel has the label of their true class. There are four possible outcomes for each voxel when comparing the predicted class to the ground truth of that class. For class c_i these are:

- True positive (TP): The network predicts that the voxel is part of class c_i. The voxel is labelled c_i in the ground truth.
- False positive (FP): The network predicts that the voxel is part of class c_i. The voxel is **not** labelled c_i in the ground truth.
- True negative (TN): The network predicts that the voxel is **not** part of class c_i. The voxel is **not** labelled c_i in the ground truth.
- False negative (FN): The network predicts that the voxel is **not** part of class c_i. The voxel is labelled c_i in the ground truth.

These are used to create metrics to evaluate the segmentations. Accuracy is a metric that can be seen as how much of the whole input volume was predicted correctly and is defined by equation 3.14. Where TP, TN, FP, FN represents the sum of voxels for each combination.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.14}$$

Sensitivity is a metric that only looks at how many voxels were correctly labelled as part of a class c_i, disregarding the voxels that are not. The sensitivity metric is given by equation 3.15.

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.15}$$

Specificity is a metric that only looks at the voxels in the ground truth not labelled class c_i and how well the network predicts these negatives. The specificity metric is defined by

equation 3.16.

$$Specifivity = \frac{TN}{TN + FP} \tag{3.16}$$

These metrics are however not always useful for comparing 3D segmentations. An example is coronary artery segmentation, where the amount of voxels that are labelled as arteries in the ground truth only makes up 2% of the volume. This means that predicting all voxels as background will give an accuracy score of 0.98. The sensitivity and specificity metrics does not penalize false positives and false negatives respectively, which means that a network prediction containing a large amount spurious responses for arteries will not make any change in the sensitivity metric, and only a small change in specificity as the amount of true negatives are much larger than false positives. A metric that is better suited for this task is the Dice similarity coefficient (DSC) (8), which looks at the true positives and penalizes for false positives and false negatives. This metric is given by equation 3.17.

$$DSC = \frac{2TP}{2TP + FP + FN} \tag{3.17}$$

Chapter 4

Results

In this chapter the results of the proposed methods are presented. The results related to coronary artery segmentation are first presented, followed by results from brain tumor segmentation. Lastly, the results from the digital rocks segmentation are presented. All training and validation DSC plots were created by using plotting functionality provided by the DeepMedic software framework (21). All 3D models were created and visualized with 3D Slicer (38).

4.1 Coronary Arteries

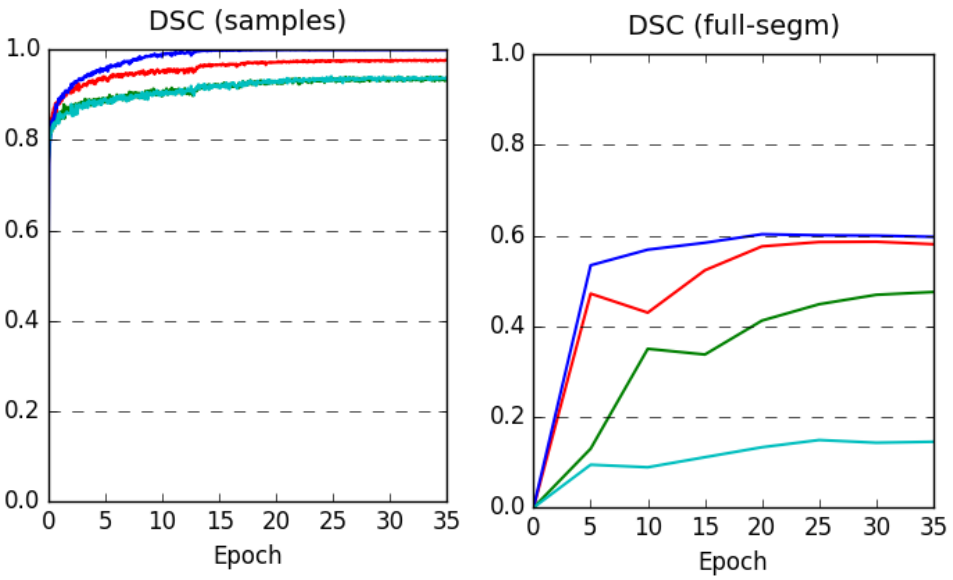
In this section the results of the coronary artery segmentation, aorta segmentation and the evaluation on clinical data is presented.

4.1.1 Coronary artery segmentation

Table 4.1 shows the DSC for training and validation of the network trained on coronary artery segmentation. The training and validation DSC for each epoch during training is shown by the plot in figure 4.1. Figures 4.2-4.6 shows 3D models of the generated ground truth as well as the segmentation results from $CoronaryCNN$, $CoronaryCNN_{0.80mm}$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$.

Experiment	DSC Training	DSC Validation
$CoronaryCNN$	0.9761	0.5812
$CoronaryCNN_{0.80mm}$	0.9996	0.5975
$CoronaryCNN_{flip}$	0.9347	0.4758
$CoronaryCNN_{ROI}$	0.9365	0.1453

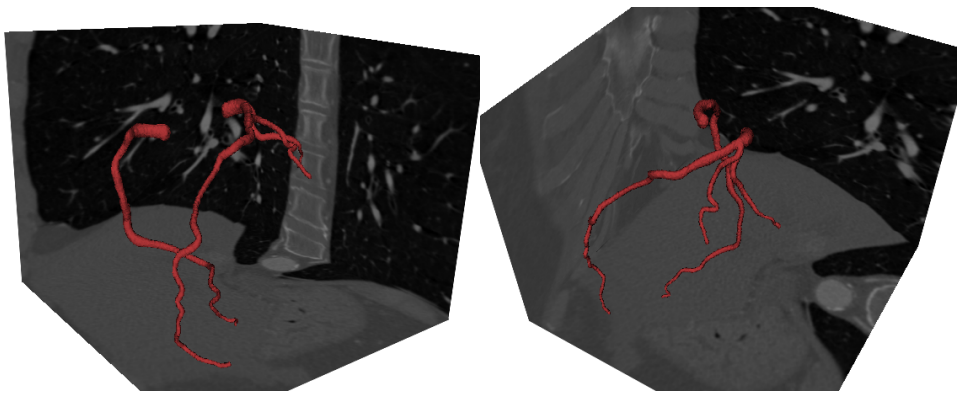
Table 4.1: Results of training and validation on the coronary artery segmentation experiments.



(a) Training

(b) Validation

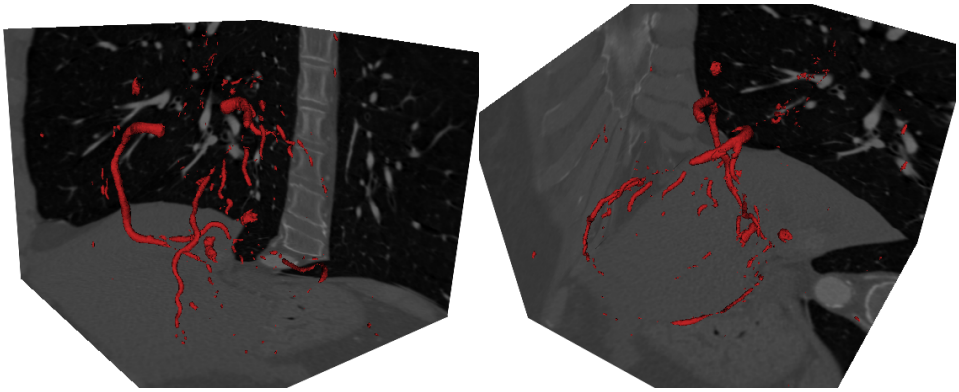
Figure 4.1: Plots of DSC of training and validation of the coronary segmentation experiments. $CoronaryCNN$ is shown in red, $CoronaryCNN_{0.80mm}$ is shown in blue, $CoronaryCNN_{flip}$ is shown in green and $CoronaryCNN_{ROI}$ is shown in cyan. The DSC on training for $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ are overlapping.



(a)

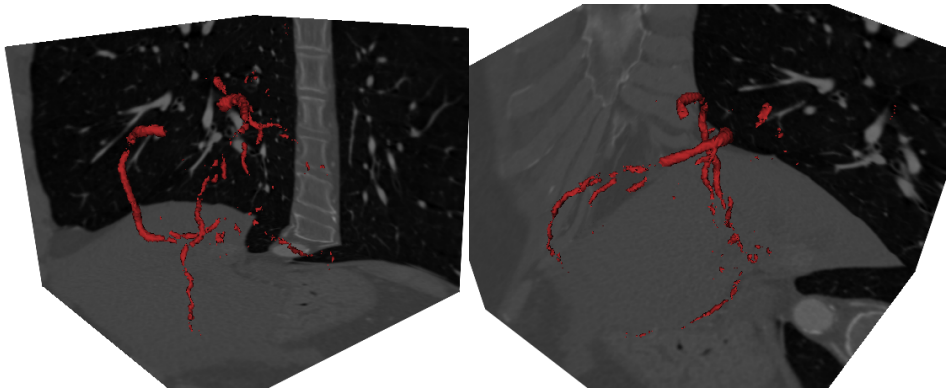
(b)

Figure 4.2: 3D models of the coronary artery ground truth generated from reference centerlines provided by the Rotterdam evaluation framework (41). The left and right images show the same model from two different angles.



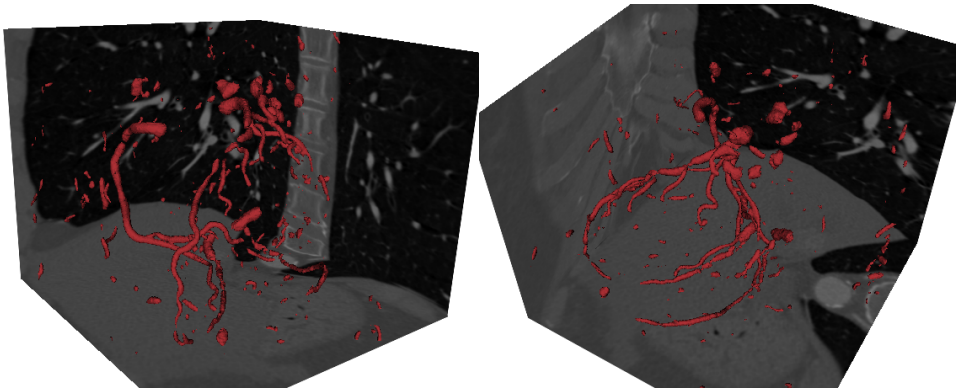
(a) **(b)**

Figure 4.3: 3D model of the coronary artery segmentation output of $CoronaryCNN$. The left and right images show the same model from two different angles. The network is able to segment parts of the coronary artery but leaves large gaps in the segmentation.



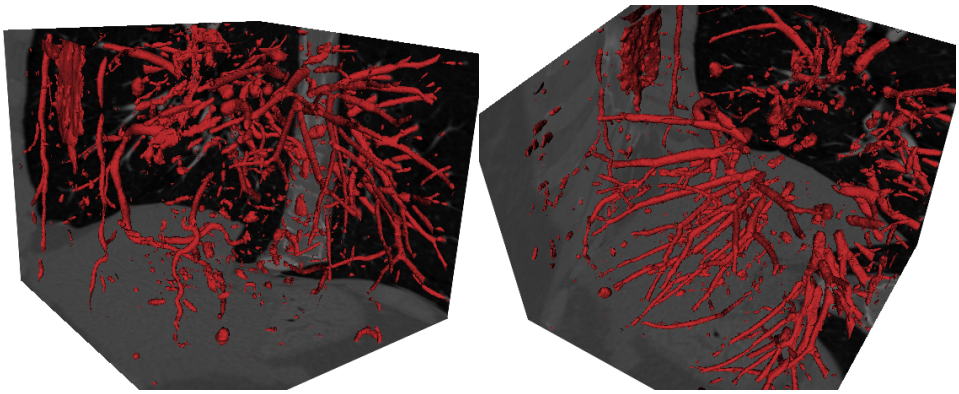
(a) **(b)**

Figure 4.4: 3D model of the coronary artery segmentation output of $CoronaryCNN_{0.80mm}$. The left and right images show the same model from two different angles. The network is able to segment parts of the coronary artery but leaves large gaps in the segmentation.



(a) **(b)**

Figure 4.5: 3D model of the coronary artery segmentation output of $CoronaryCNN_{flip}$. The left and right images show the same model from two different angles. The ntwork produces less gaps but also introduces more spurious responses.



(a) **(b)**

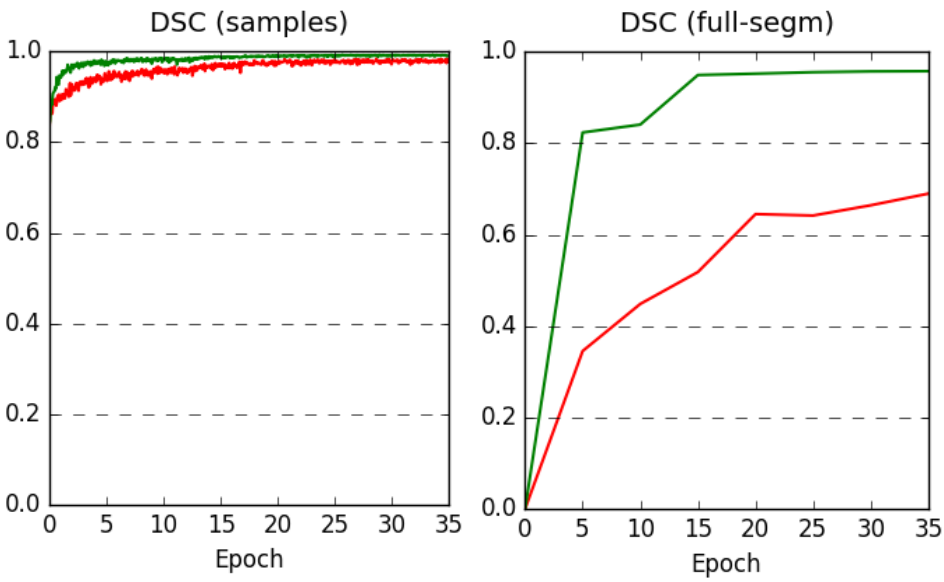
Figure 4.6: 3D model of the coronary artery segmentation output of $CoronaryCNN_{ROI}$. The left and right images show the same model from two different angles. The network segments a large amount of spurious responses (mostly tubular structures) in addition to the coronary arteries.

4.1.2 Aorta segmentation

Table 4.2 shows the training and validation DSC for the networks trained to segment the aorta from CT images. The training and validation DSC for each epoch during training is shown by the plots in figure 4.7. Figure 4.8 shows 3D models created from the segmentation output of both $AortaCNN_{0.40mm}$ and $AortaCNN_{0.80mm}$ for all four validation volumes. The ground truth for each validation volume is also shown in the figure.

Experiment	DSC Training	DSC Validation
$AortaCNN_{0.40mm}$	0.9796	0.6901
$AortaCNN_{0.80mm}$	0.9912	0.9579

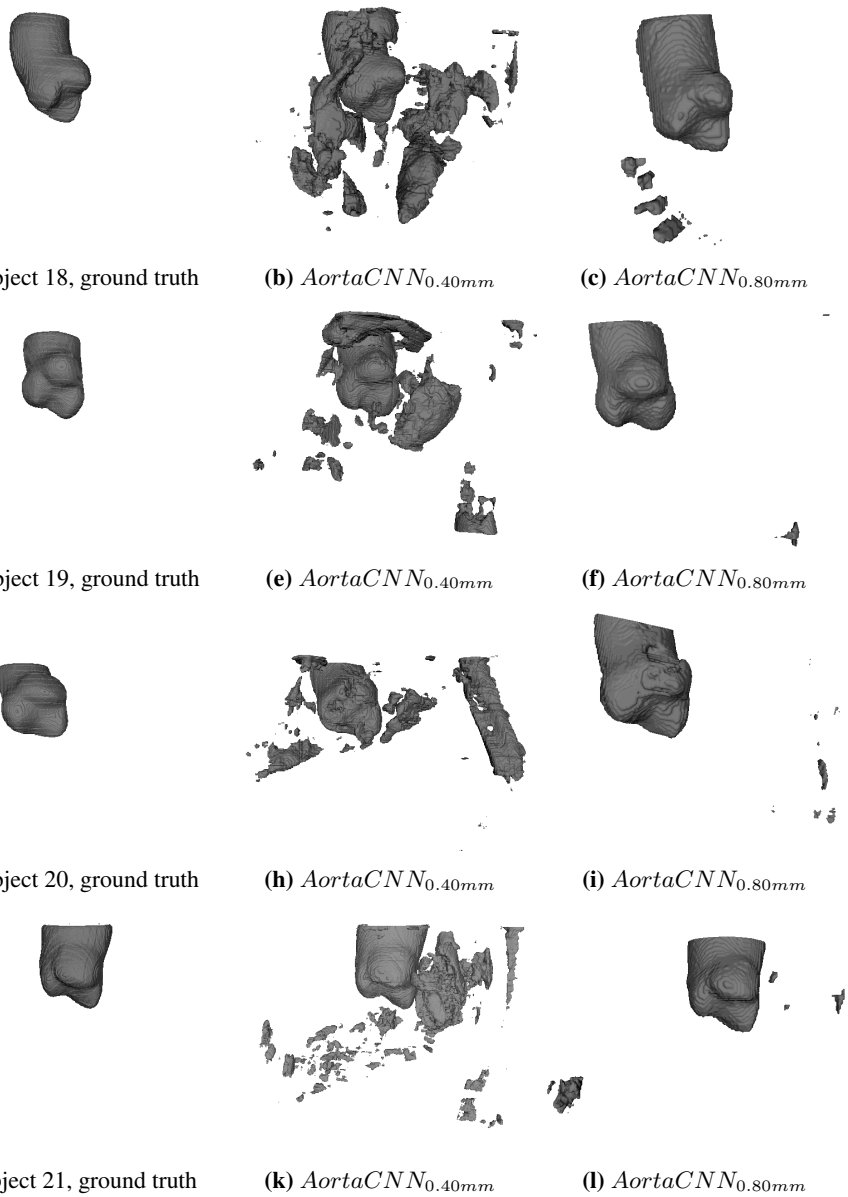
Table 4.2: Results of training and validation on aorta segmentation experiments.



(a) Training

(b) Validation

Figure 4.7: Plot of training and validation DSC for $AortaCNN_{0.40mm}$ shown in red and $AortaCNN_{0.80mm}$ shown in green.



(a) Subject 18, ground truth **(b)** $AortaCNN_{0.40mm}$ **(c)** $AortaCNN_{0.80mm}$

(d) Subject 19, ground truth **(e)** $AortaCNN_{0.40mm}$ **(f)** $AortaCNN_{0.80mm}$

(g) Subject 20, ground truth **(h)** $AortaCNN_{0.40mm}$ **(i)** $AortaCNN_{0.80mm}$

(j) Subject 21, ground truth **(k)** $AortaCNN_{0.40mm}$ **(l)** $AortaCNN_{0.80mm}$

Figure 4.8: Results of aorta segmentation. Left column shows the ground truth for each subject. Middle column shows the segmentation result of experiment $AortaCNN_{0.40mm}$: resampling to 0.40mm voxel spacing. These images show large amount of oversegmentation. Right column shows the result of experiment $AortaCNN_{0.80mm}$: resampling to 0.80mm voxel spacing. These images show only a few spurious responses.

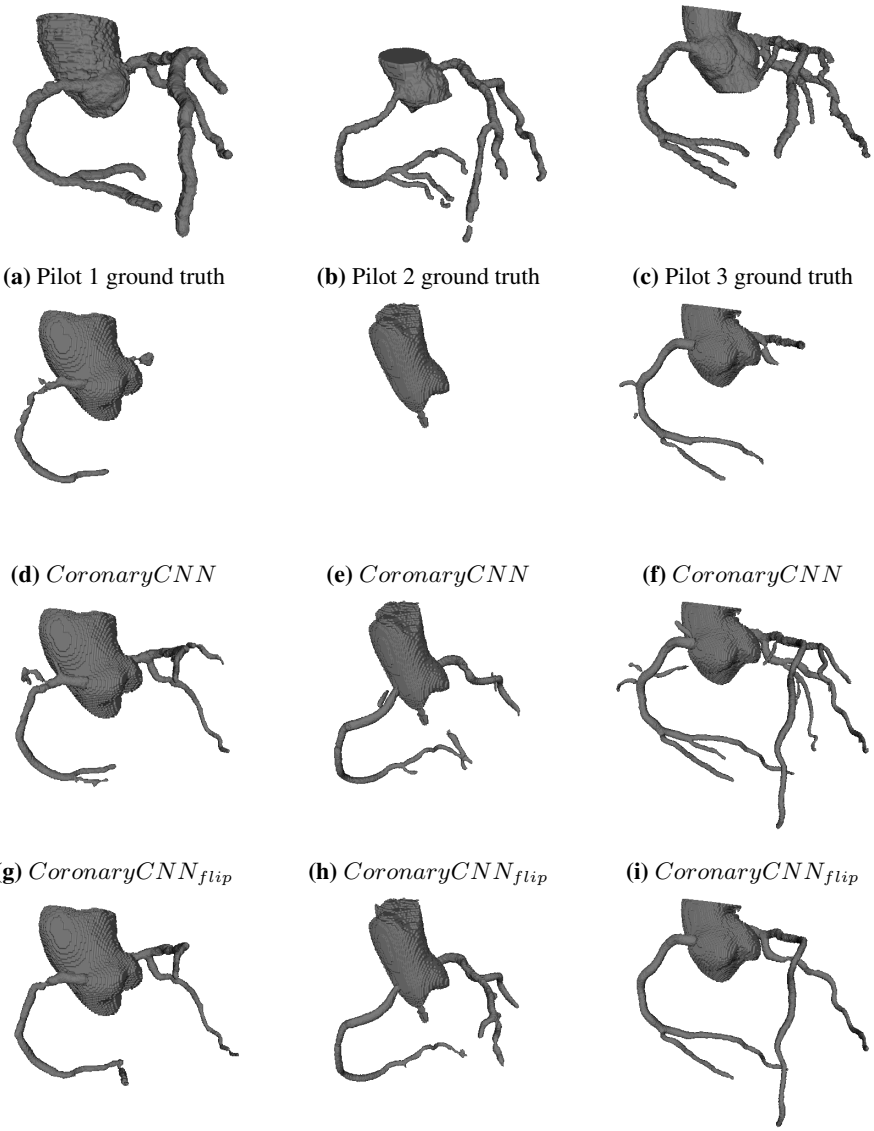
4.1.3 Evaluation on pilot dataset

The pilot dataset provided by St. Olavs Hospital includes three CT volumes with manual segmentations performed by a clinical expert. The manual segmentations contain both the aorta and the coronary arteries. Both the aorta and coronary arteries are extracted and refined using the method proposed for extracting the coronary arteries. The aorta and coronary artery results are then combined into one volume and evaluated against the ground truth. Three versions of the proposed method are created by using $CoronartyCNN$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ to segment the coronary arteries.
$AortaCNN_{0.80mm}$ is used to segment the aorta for all the tests, meaning that all tests on each pilot volume will share the same aorta segmentation. The test DSC from evaluation using $CoronaryCNN$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ is shown in table 4.3.

3D models of the ground truth for the pilot volumes and the segmentation produced by the tests are shown in figure 4.9.

Test volume	$CoronaryCNN$	$CoronaryCNN_{flip}$	$CoronaryCNN_{ROI}$
Pilot 1	0.8434	0.8550	0.8551
Pilot 2	0.5820	0.6394	0.6529
Pilot 2	0.8280	0.8405	0.8382
Mean DSC	0.7512	0.7783	0.7820

Table 4.3: Test DSC of three versions of the proposed method for extracting coronary arteries evaluated on the pilot dataset provided by St. Olavs Hospital. The coronary artery and aorta segmentation results are combined into one volume before evaluation. $AortaCNN_{0.80mm}$ is used to segment the aorta in all tests.



(a) Pilot 1 ground truth **(b)** Pilot 2 ground truth **(c)** Pilot 3 ground truth

(d) $CoronaryCNN$ **(e)** $CoronaryCNN$ **(f)** $CoronaryCNN$

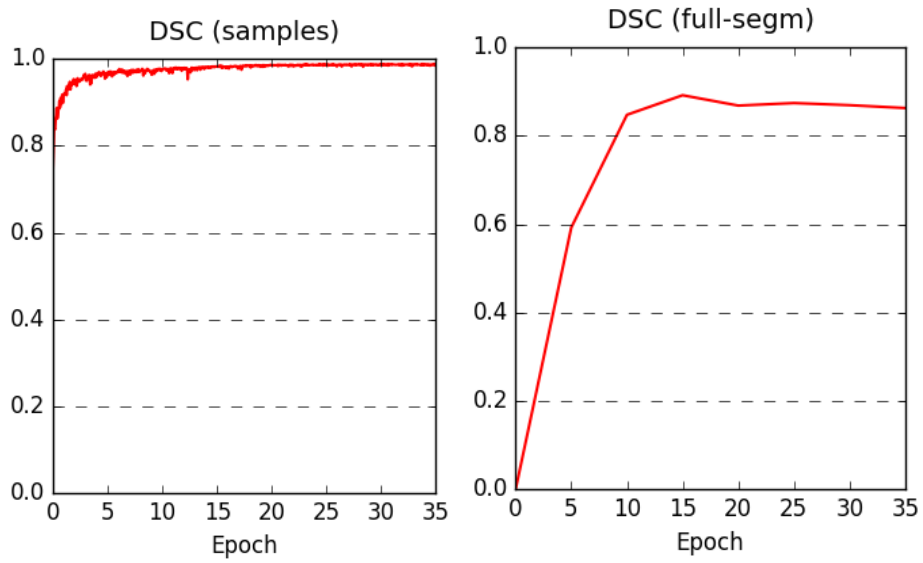
(g) $CoronaryCNN_{flip}$ **(h)** $CoronaryCNN_{flip}$ **(i)** $CoronaryCNN_{flip}$

(j) $CoronaryCNN_{ROI}$ **(k)** $CoronaryCNN_{ROI}$ **(l)** $CoronaryCNN_{ROI}$

Figure 4.9: Evaluation on pilot dataset. The left, middle and right column show the ground truth and segmentation results of pilot 1, pilot 2 and pilot 3 CT volumes respectively. The first row show the manual segmentations performed by a clinical expert. The next rows show segmentation results of $CoronaryCNN$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ refined by using connected component analysis to discard responses not connected to the aorta. $AortaCNN_{0.80mm}$ was used to segment the aorta in each volume.

4.2 Brain tumor segmentation

Figure 4.10 shows plots of the DSC for training and validation for BrainTumorCNN. After 15 epochs it can be observed that the validation DSC starts to decrease while the training DSC still increases, most likely due to overfitting. A technique known as early stopping is used by treating the trained model at 15 epochs as the final model and discarding the subsequent models. Table 4.4 shows the validation DSC and inference time on the four validation volumes. Qualitative results of the segmentation are shown in the form of slices in figure 4.11. Figures 4.12-4.15 show 3D models of the segmentation results and corresponding ground truth.



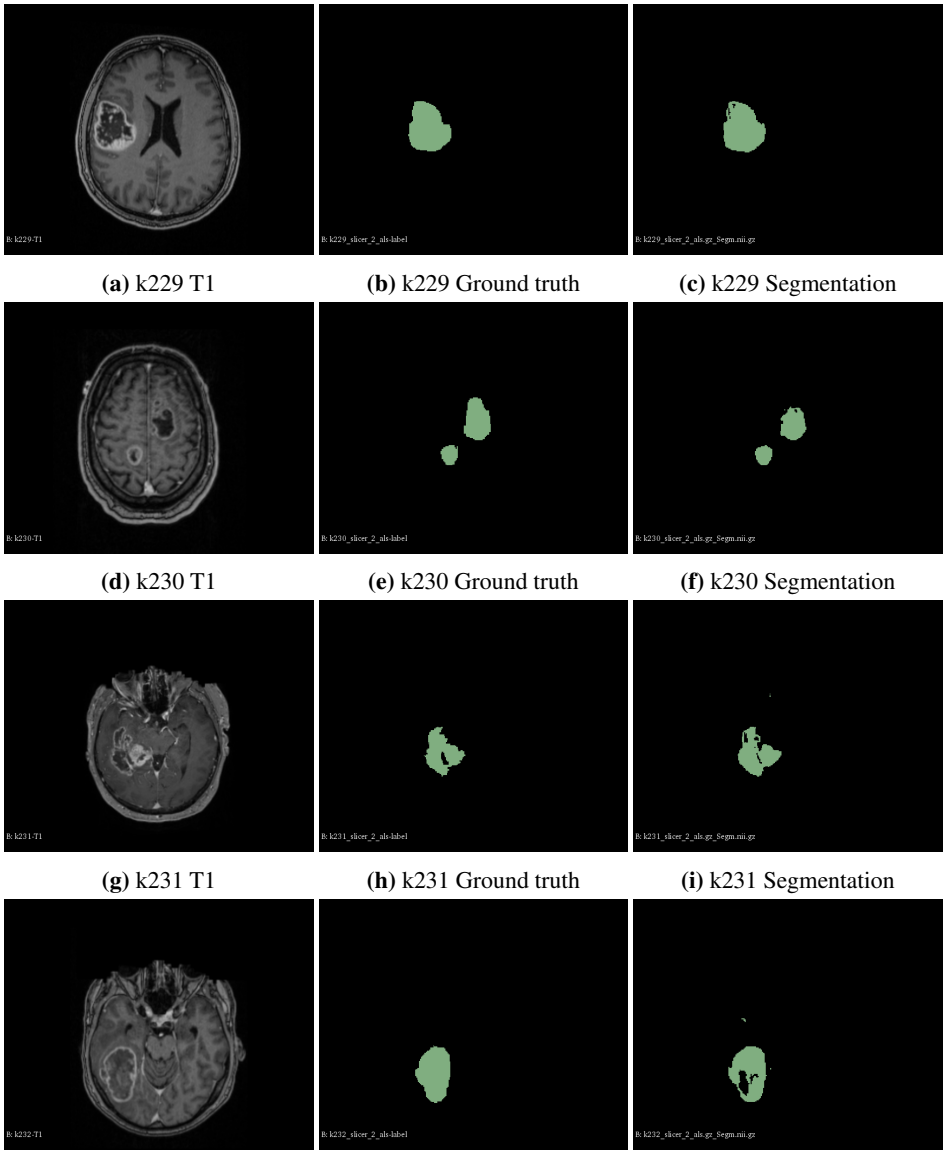
(a) Training

(b) Validation

Figure 4.10: Plot of training and validation DSC for glioblastoma segmentation.

Subject	Validation DSC
k229	0.9249
k230	0.8301
k231	0.9265
k232	0.8872
Mean DSC	0.8922

Table 4.4: Validation DSC and inference time for the four validation volumes.



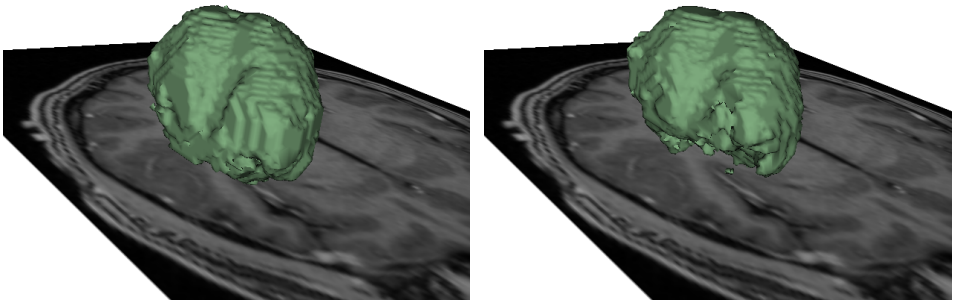
(a) k229 T1 **(b)** k229 Ground truth **(c)** k229 Segmentation

(d) k230 T1 **(e)** k230 Ground truth **(f)** k230 Segmentation

(g) k231 T1 **(h)** k231 Ground truth **(i)** k231 Segmentation

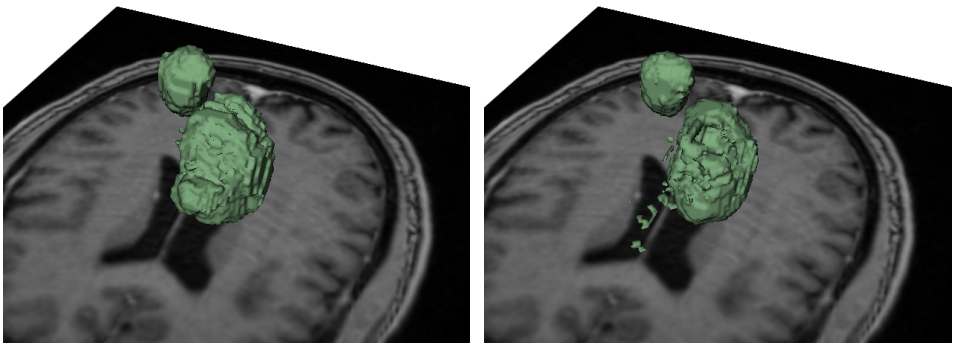
(j) k232 T1 **(k)** k232 Ground truth **(l)** k232 Segmentation

Figure 4.11: Slices showing the result of the model trained on brain tumor segmentation for subjects k229, k230, k231 and k232. The segmentation is able to large parts of the tumors, but contains some spurious responses.



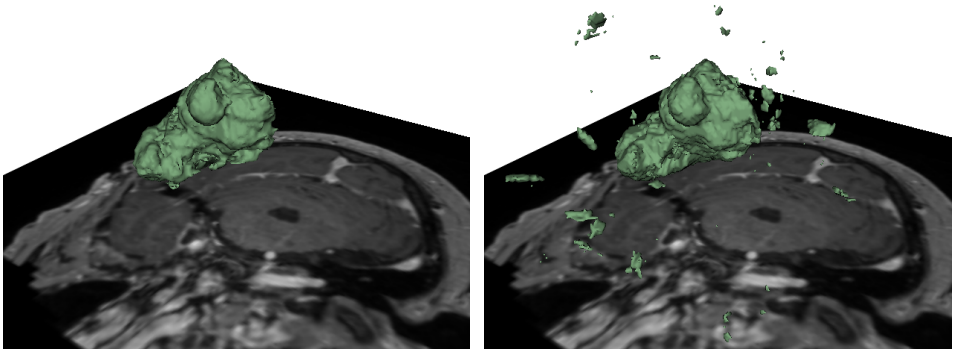
(a) Ground truth **(b)** Segmentation

Figure 4.12: 3D models of brain tumor on subject k229. The segmentation is missing some parts found in the ground truth.



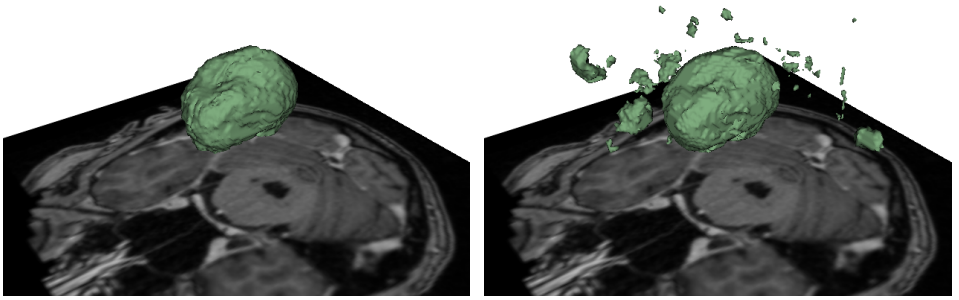
(a) Ground truth **(b)** Segmentation

Figure 4.13: 3D models of brain tumor on subject k230. The segmentation is missing some parts found in the ground truth.



(a) Ground truth **(b)** Segmentation

Figure 4.14: 3D models of brain tumor on subject k231. The segmentation segments most of the tumor but also contains spurious responses.



(a) Ground truth **(b)** Segmentation

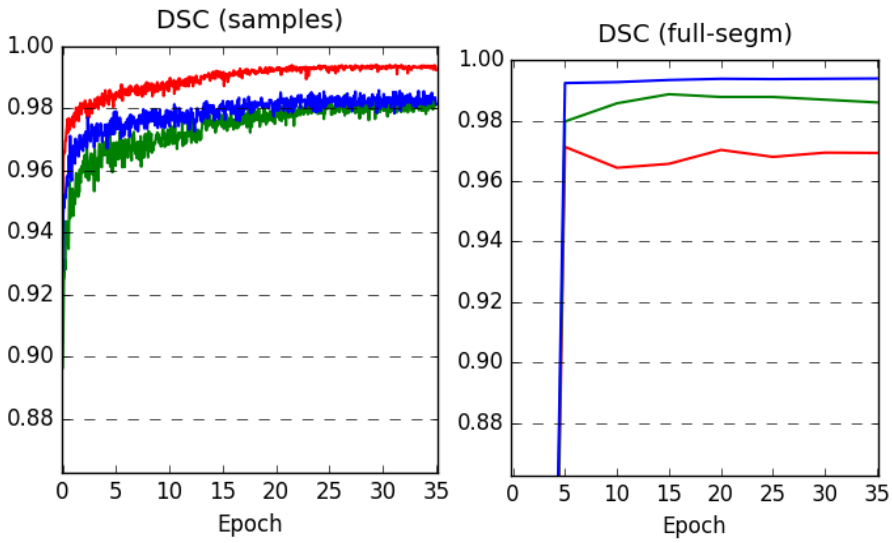
Figure 4.15: 3D models of brain tumor on subject k232. The segmentation segments most of the tumor but also contains spurious responses.

4.3 Digital rock segmentation

Figures 4.16-4.18 show plots of the training and validation DSC on pore, multi phase and grain segmentation using the networks $RockCNN_1$, $RockCNN_{10}$ and $RockCNN_{40}$. These networks were trained on different amount of subvolumes from the bentheimer sandstone volume. Figures 4.19-4.19 show the same type of plots but with $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$ instead. The training and validation DSC for all networks are summarized in table 4.5. The results for $RockCNN_{BT}$ are left out as these are the same as the results of $RockCNN_{40}$.

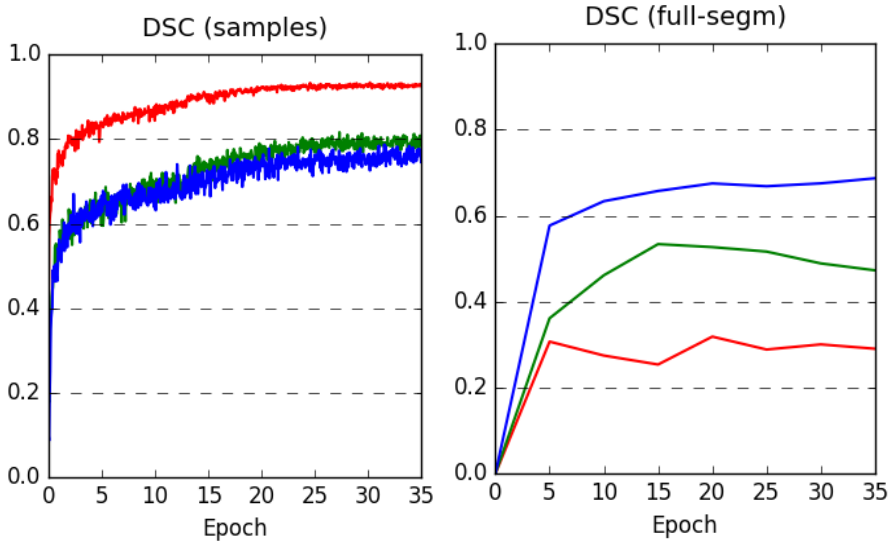
The test DSC for experiments on training and testing on different types of rock are shown in tables 4.6-4.9.

Figure 4.22 show a slice of a bentheimer sandstone subvolume, corresponding ground truth and segmentation results of $RockCNN_1$, $RockCNN_{10}$ and $RockCNN_{40}$. A similar comparison of segmentation results and ground truth for $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$ is given by figure 4.23.



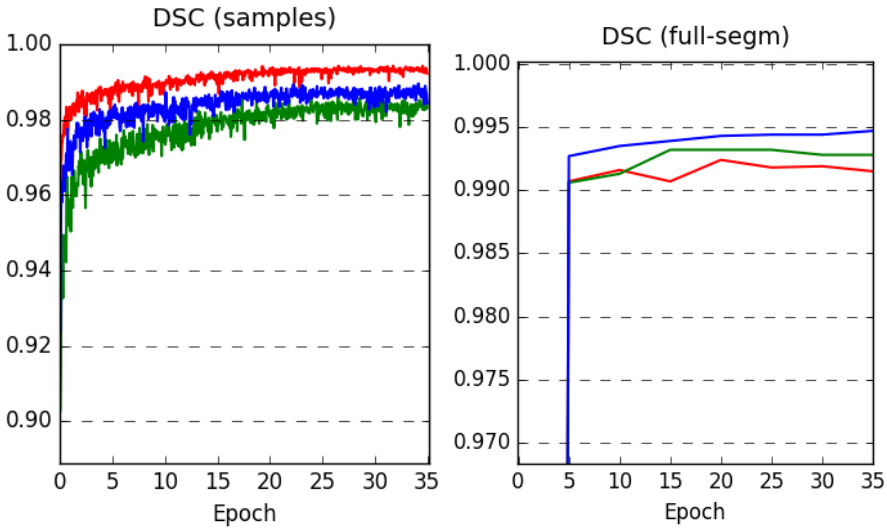
(a) Training **(b)** Validation

Figure 4.16: Training and validation plots for **pore** segmentation experiments $RockCNN_1$ shown in red, $RockCNN_{10}$ shown in green and $RockCNN_{40}$ shown in blue.



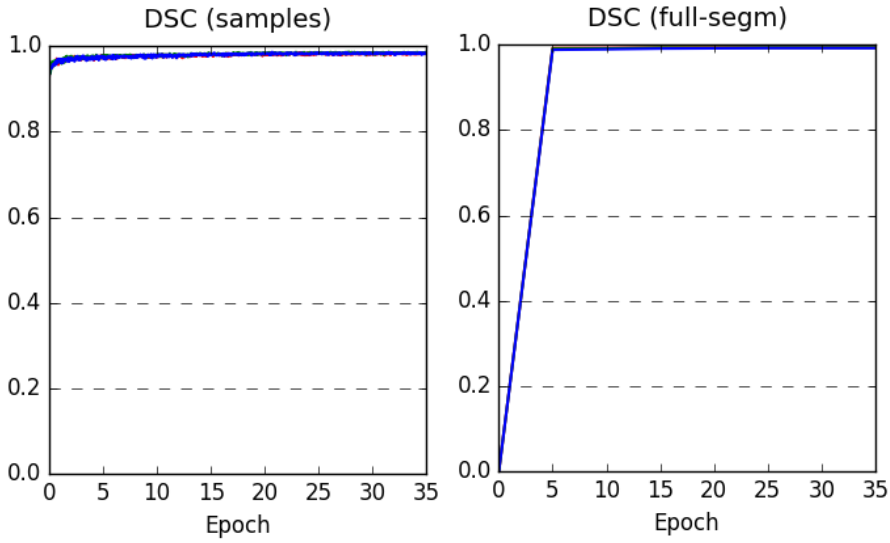
(a) Training **(b)** Validation

Figure 4.17: Training and validation plots for **multi phase** segmentation experiments $RockCNN_1$ shown in red, $RockCNN_{10}$ shown in green and $RockCNN_{40}$ shown in blue.



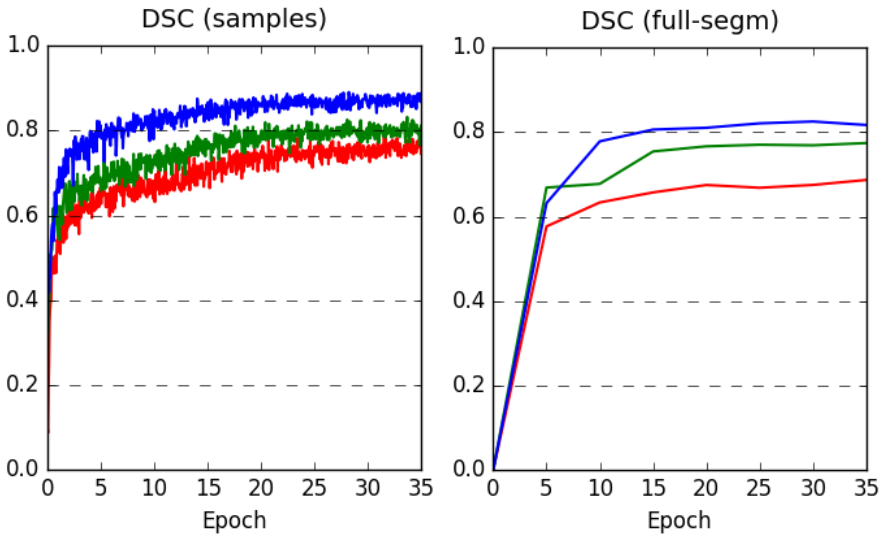
(a) Training **(b)** Validation

Figure 4.18: Training and validation plots for **grain** segmentation experiments $RockCNN_1$ shown in red, $RockCNN_{10}$ shown in green and $RockCNN_{40}$ shown in blue.



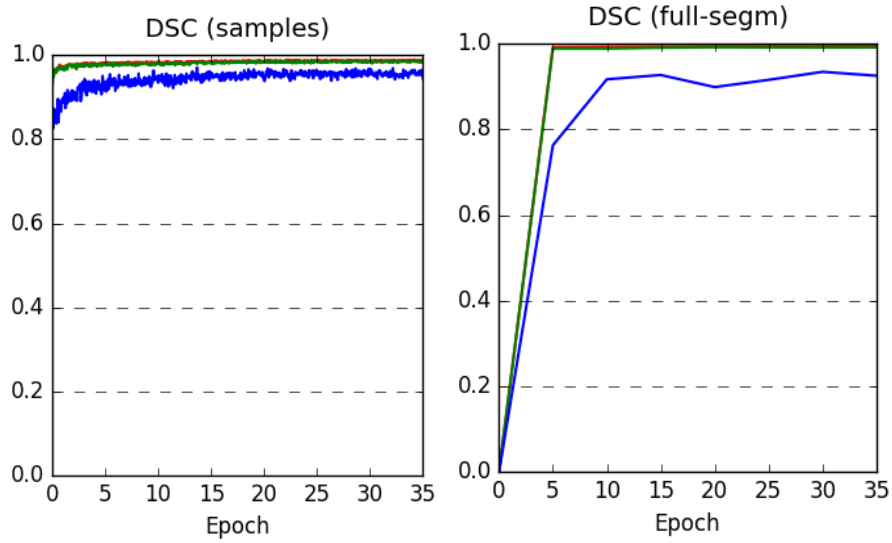
(a) Training **(b)** Validation

Figure 4.19: Training and validation plots for **pore** segmentation experiments $RockCNN_{BT}$ shown in red, $RockCNN_{BT+BR}$ shown in green and $RockCNN_{BT+BR+CA}$ shown in blue.



(a) Training **(b)** Validation

Figure 4.20: Training and validation plots for **multi phase** segmentation experiments $RockCNN_{BT}$ shown in red, $RockCNN_{BT+BR}$ shown in green and $RockCNN_{BT+BR+CA}$ shown in blue.



(a) Training **(b)** Validation

Figure 4.21: Training and validation plots for **grain** segmentation experiments $RockCNN_{BT}$ shown in red, $RockCNN_{BT+BR}$ shown in green and $RockCNN_{BT+BR+CA}$ shown in blue.

	DSC Training			DSC Validation		
Experiment	**Pore**	**Multi phase**	**Grain**	**Pore**	**Multi phase**	**Grain**
$RockCNN_1$	0.9933	0.9273	0.9933	0.9693	0.2907	0.9915
$RockCNN_{10}$	0.9809	0.7935	0.9839	0.9860	0.4731	0.9928
$RockCNN_{40}$	0.9828	0.7624	0.9872	0.9940	0.6387	0.9952
$RockCNN_{BT}$	-	-	-	-	-	-
$RockCNN_{BT+BR}$	0.9843	0.8018	0.9849	0.9939	0.7745	0.9919
$RockCNN_{BT+BR+CA}$	0.9835	0.8724	0.9564	0.9924	0.8173	0.9257

Table 4.5: Training and validation results.

Experiment	**DSC Pore**	**DSC Multi phase**	**DSC Grain**
$RockCNN_{BT}$	0.9946	0.6922	0.9956
$RockCNN_{BT+BR}$	0.9935	0.6434	0.9954
$RockCNN_{BT+BR+CA}$	0.9921	0.5889	0.9949

Table 4.6: Test DSC scores for type 1, bentheimer sandstone

Experiment	**DSC Pore**	**DSC Multi phase**	**DSC Grain**
$RockCNN_{BT}$	0.9789	0.4060	0.9751
$RockCNN_{BT+BR}$	0.9942	0.8558	0.9888
$RockCNN_{BT+BR+CA}$	0.9937	0.8467	0.9880

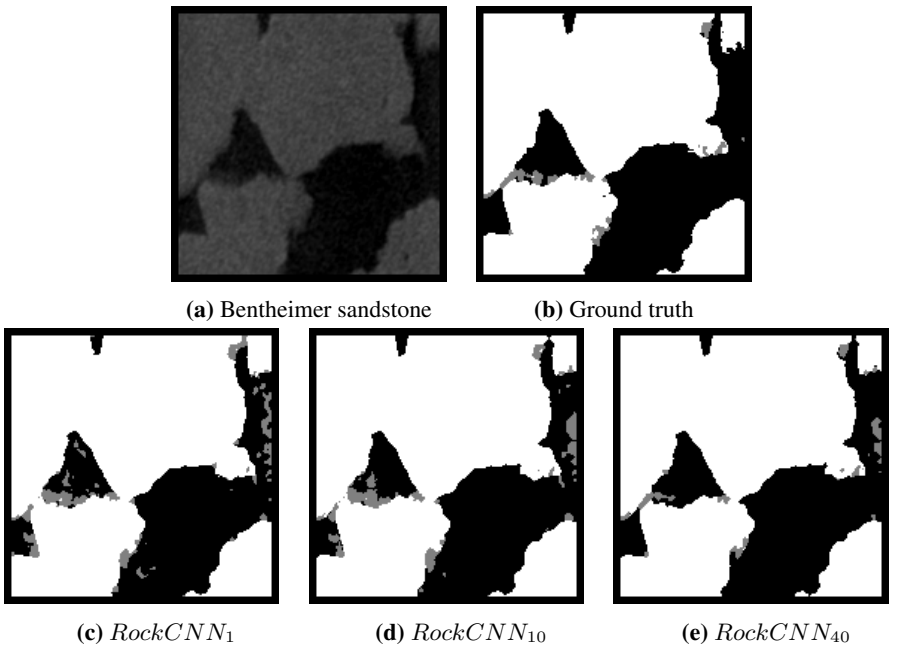
Table 4.7: Test DSC scores for type 2, berea sandstone

Experiment	**DSC Pore**	**DSC Multi phase**	**DSC Grain**
$RockCNN_{BT}$	0.9866	0.1229	0.5048
$RockCNN_{BT+BR}$	0.9895	0.2146	0.5164
$RockCNN_{BT+BR+CA}$	0.9940	0.9117	0.8260

Table 4.8: Test DSC scores for type 1, carbonate

Experiment	**DSC Pore**	**DSC Multi phase**	**DSC Grain**
$RockCNN_{BT}$	0.9867	0.4070	0.8252
$RockCNN_{BT+BR}$	0.9924	0.5713	0.8335
$RockCNN_{BT+BR+CA}$	0.9933	0.7824	0.9363

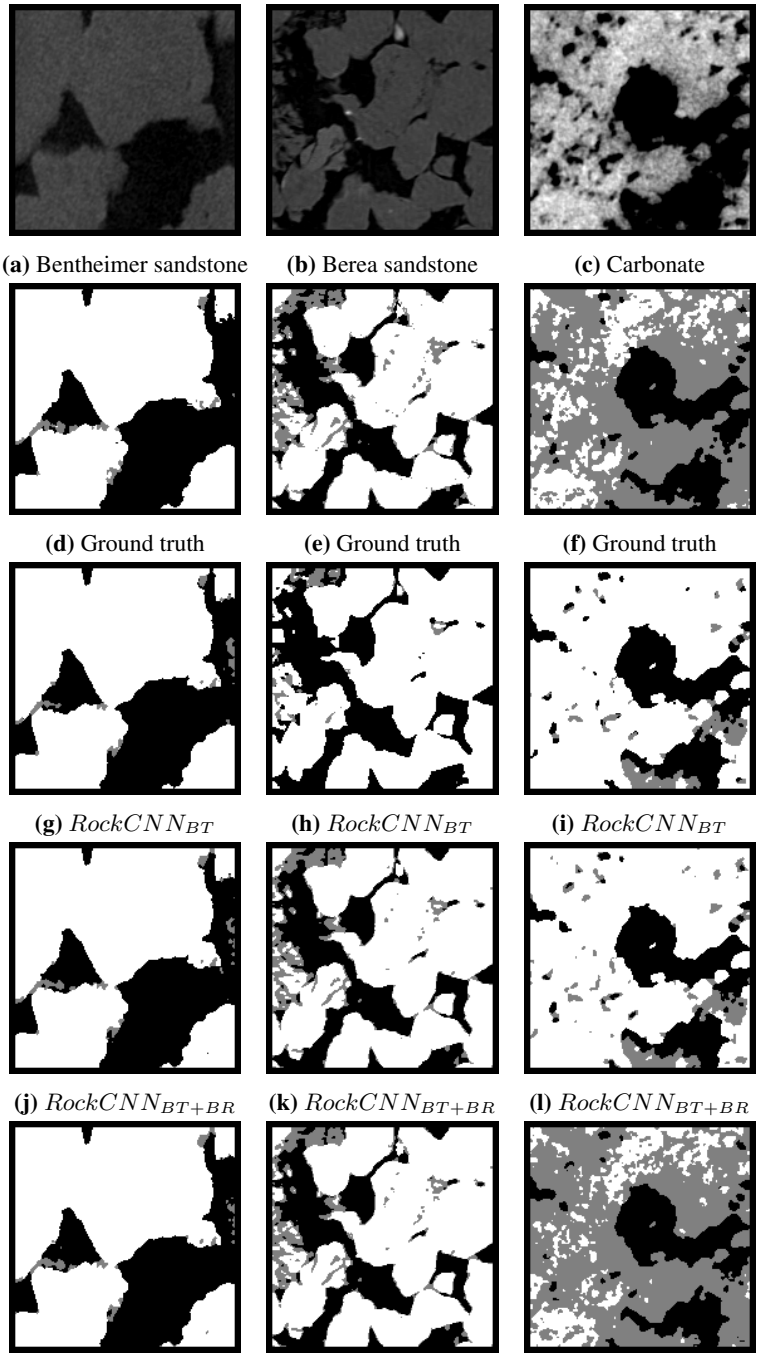
Table 4.9: Mean DSC across all three types.



(a) Bentheimer sandstone **(b)** Ground truth

(c) $RockCNN_1$ **(d)** $RockCNN_{10}$ **(e)** $RockCNN_{40}$

Figure 4.22: Results from experiments $RockCNN_1$, $RockCNN_{10}$ and $RockCNN_{40}$. For the ground truth and segmentation results, pore is represented as black, multi phase as gray and grain as white.



(a) Bentheimer sandstone **(b)** Berea sandstone **(c)** Carbonate

(d) Ground truth **(e)** Ground truth **(f)** Ground truth

(g) $RockCNN_{BT}$ **(h)** $RockCNN_{BT}$ **(i)** $RockCNN_{BT}$

(j) $RockCNN_{BT+BR}$ **(k)** $RockCNN_{BT+BR}$ **(l)** $RockCNN_{BT+BR}$

(m) $RockCNN_{BT+BR+CA}$ **(n)** $RockCNN_{BT+BR+CA}$ **(o)** $RockCNN_{BT+BR+CA}$

Figure 4.23: Comparison of the segmentations by the networks. For the ground truth and segmentation results, pore is represented as black, multi phase as gray and grain as white.

4.4 Timing

The training and inference times of the different networks are presented in table 4.10.

Experiment	Training time	Mean inference time per volume
$CoronaryCNN$	71254s ≈ 20h	285s ≈ 5m
$CoronaryCNN_{0.80mm}$	28691s ≈ 8h	40s
$CoronaryCNN_{flip}$	69714s ≈ 19h	289s ≈ 5m
$CoronaryCNN_{ROI}$	70332s ≈ 20h	290s ≈ 5m
$AortaCNN_{0.40mm}$	67865s ≈ 19h	285s ≈ 5m
$AortaCNN_{0.80mm}$	28721s ≈ 8h	41s
$BrainTumorCNN$	28356s ≈ 8h	55s
$RockCNN_{1}$	28504s ≈ 8h	30s
$RockCNN_{10}$	28986s ≈ 8h	28s
$RockCNN_{40}$	28599s ≈ 8h	28s
$RockCNN_{BT+BR}$	28794s ≈ 8h	26s
$RockCNN_{BT+BR+CA}$	28668s ≈ 8h	30s

Table 4.10: Training and mean inference times for the networks trained in this thesis. The inference time of the networks used for aorta and coronary artery segmentation were obtained by evaluating them on the pilot dataset. The table shows that networks trained on smaller volumes have similar training and inference times. Networks trained on larger volumes use on average 11 to 12 hours more for training.

Chapter 5

Discussion

In this chapter the results are discussed. The strengths and weaknesses of the chosen method are discussed based on the results on coronary artery segmentation, brain tumor segmentation and segmentation on the rock dataset.

5.1 Coronary Artery Segmentation

In this section the results of the networks trained on aorta segmentation and coronary segmentation are discussed, followed by a discussion of the proposed method for refining the segmentation.

5.1.1 Results of network trained on coronary arteries

The output segmentation of the networks $CoronaryCNN$, $CoronaryCNN_{0.80mm}$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ shows that the network is able to segment parts of the coronary arteries. The first experiment, only using preprocessing and resampling to 0.40mm voxel spacing, gives a training DSC of 0.9761 and a validation DSC of 0.5812 as shown in table 4.1. This means that the network performs much better on the samples it has been trained on, indicating that the model is being overfitted on the training data.

The network $CoronaryCNN_{0.80mm}$, trained with volumes resampled to 0.80mm voxel spacing, shows clearer signs of overfitting. The validation DSC is similar to the validation DSC for $CoronaryCNN$, but the training DSC rises to 0.996, meaning that it is able to segment the training samples almost perfectly. When resampling to 0.80mm voxel spacing, the input volume is essentially downscaled by a factor of 2 in each dimension compared to the volumes with 0.40mm voxel spacing. This leads to available amount of voxels in the training data is reduced by a factor of 8. Due to large amount of weights in the 3D CNN architecture it is possible for the network to learn a direct mapping between the input samples and the corresponding correct output segmentation, which performs poorly

at generalizing as shown by the validation DSC.

To try to reduce the amount of overfitting, data augmentation was used for the networks $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$. The data was augmented by mirroring the sampled patches across each axis randomly with an equal probability. This will essentially increase the amount of unique data that the network will see during training by 8 times. The plot of the training DSC and the validation DSC shown in figure 4.1 shows that both training and validation DSC for $CoronaryCNN_{flip}$ are lower than $CoronaryCNN$ and $CoronaryCNN_{0.80mm}$. This means that the network has more difficulty in correctly segmenting the training samples. The decrease in validation DSC for $CoronaryCNN_{flip}$ in comparison to $CoronaryCNN$ and $CoronaryCNN_{0.80mm}$ can be explained by their segmentations shown in figure 4.3 and 4.5. The segmentation produced by $CoronaryCNN_{flip}$ contains more of the coronary tree than the segmentation produced by $CoronaryCNN$ when comparing against the ground truth in figure 4.2. However the segmentation by $CoronaryCNN_{flip}$ contains a larger amount of spurious responses while the segmentation by $CoronaryCNN$ contains far less. This shows that the spurious responses has a larger effect on the DSC than the correct predictions.
While $CoronaryCNN_{flip}$ is able to segment more of the coronary tree it still leaves gaps in the output segmentation seen in figure 4.5.

For $CoronaryCNN_{ROI}$, a region of interest around the coronary artery vessels were created and used for training. The experiment also utilizes the same data augmentation as $CoronaryCNN_{flip}$. As seen in figure 4.1 the training DSC of $CoronaryCNN_{ROI}$ overlaps with the training DSC of $CoronaryCNN_{flip}$. The training DSC is based on the samples only. The samples for $CoronaryCNN_{flip}$ are selected with a 50% probability of being centered on a coronary artery voxel or being centered on a background voxel found anywhere in the volume. With $CoronaryCNN_{ROI}$ the voxels are still selected with a 50% probability, but the background voxels are only selected from a small area around the coronary arteries. This means that the a larger variation in negative samples ($CoronaryCNN_{flip}$) has little impact on the training DSC. During validation the ROI is not used and the whole volume is segmented by the network, leading to a severe decrease in validation DSC compared to $CoronaryCNN_{flip}$. Figure 4.6 shows the segmentation result of $CoronaryCNN_{ROI}$. The segmentation contains a lot more responses than the segmentation from $CoronaryCNN_{flip}$. The oversegmentation is not completely random and it is biased to segment mostly tubular structures. This also means that the networks in the other experiments has learned to extract tubular structures in only a specific area around the heart.

The experiments gives some insight into the behaviour of the trained networks, but due to small amounts of training data more cross validation needs to be performed to see how well the network will generalize on new data. A possibility exist that the validation volume that is used is an easier or harder case than the other volumes. The cross validation consists of training several networks with the same configurations but changing which volumes are used for training and which volume is used for validation. The scores from cross validation are then averaged to obtain a statistically better metric.

5.1.2 Result on aorta segmentation

For aorta segmentation two experiments were carried out to observe the effect of different scales of input data. The experiments with $AortaCNN_{0.40mm}$ and $AortaCNN_{0.80mm}$ are equal, with the difference being that they are resampled to 0.40mm and 0.80mm voxel spacing respectively. The DSC scores for training and validation of the networks are given by table 4.2 and shows that $AortaCNN_{0.80mm}$ outperforms $AortaCNN_{0.40mm}$ in both training and validation DSC. Looking at the training and validation DSC plots for $AortaCNN_{0.40mm}$ shown in figure 4.7, it can be seen that the validation DSC seems to be still increasing after 35 epochs. The network could be trained further to see if the validation DSC would still increase, but was stopped at 35 epochs for comparison purposes. The training DSC shows that the trained network performs well on the training samples and is able to learn the appearance of the aorta. The lower validation is most likely caused by the network segmenting more than just the aorta as shown in figure 4.8. The network manages to segment the aorta, but also segments other large regions not part of the aorta. The size and connectivity of these regions makes it harder to extract only the aorta from the segmentation output.

$AortaCNN_{0.40mm}$ is outperformed by $AortaCNN_{0.80mm}$, which achieves a validation DSC of 0.9579. Figure 4.7 shows that the network is able to converge after only 15 epochs, which is faster compared to $AortaCNN_{0.40mm}$. The segmentation output of $AortaCNN_{0.80mm}$ shown in figure 4.8 contains fewer responses than the results of $AortaCNN_{0.40mm}$. This means that it is possible to label the separate connected responses and select the connected component containing the largest amount of voxels to obtain the aorta.

The results of $AortaCNN_{0.40mm}$ and $AortaCNN_{0.80mm}$ shows that the same network architecture with same hyperparameters performs better on a smaller volume with larger voxel spacing for aorta segmentation. The effect of resampling to 0.80mm compared to 0.40mm is that the former will increase the receptive field in mm^3 by 8 times, essentially giving the network more spatial information to work with. It is also faster to perform training and inference when resampling to 0.80mm voxel spacing, as the amount of data is reduced 8 times compared to resampling to 0.40mm voxel spacing. An experiment consisting of keeping the 0.40mm voxel spacing but instead increasing the size of the patch used for the downsampled pathway can be tested. The downsampling factor would have to be increased to maintain the input size of the network. This could lead to better segmentation due to working on a larger part of the image while still keeping a small and accurate normal resolution pathway. However the aorta segmentation in this thesis is only used as a step to extract the coronary arteries, where voxel-perfect segmentation of the complete aorta is not necessary.

5.1.3 Results on the pilot dataset

The proposed method for refining the coronary artery segmentation works by using two separately trained networks (aorta and coronary artery segmentation) as shown in figure 3.4. The method selects only the responses in the intermediate coronary segmentation that

are connected to the segmented aorta and discards other spurious responses. Since the ground truth segmentation for the pilot dataset consists of both the aorta and the coronary arteries, the aorta and coronary artery results from the proposed method are combined into one volume for evaluation. Three versions of the proposed method are evaluated, with $CoronaryCNN$, $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ used for coronary artery segmentation. $AortaCNN_{0.80}$ is used for all tests. Table 4.3 shows the test DSC of the different versions. Figure 4.9 shows 3D models of the ground truth and segmentations of the three pilot volumes by the three versions of the method.

The test DSC for the different versions are similar in value. This is mainly due to all versions using the same network for aorta segmentation and the aorta being a large part of the ground truth voxels. An example is the DSC of the version using $CoronaryCNN$ tested on the pilot 2 volume. The network fails to extract any coronary arteries as shown in figure 4.9e, but still achieves a DSC of 0.5820. The same figure also shows that the network is segmenting a larger part of the aorta than what is provided by the manual segmentation. The versions using $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ evaluated on the pilot 2 volume are able to extract more of the coronary arteries and achieves higher DSC of 0.6394 and 0.6529 respectively. The low DSC compared to evaluations on the other pilot volumes is mainly due to the difference between the aorta segmentation produced by the method and the ground truth. The ground truth segmentations for the pilot 1 and pilot 2 volumes does not contain accurate segmentations of the aorta, shown in figures 4.9a and 4.9c, which means that comparison with the aorta segmentation of the proposed method will not be optimal.

Comparing the extracted coronary arteries of the three versions of the method, the version using $CoronaryCNN$ gives the lowest DSC for all pilot volumes. This can also be seen in figures 4.9d, 4.9e and 4.9f, where the method only manages to extract small parts of the coronary artery tree or nothing at all, as shown in figure 4.9e. The versions using $CoronaryCNN_{flip}$ and $CoronaryCNN_{ROI}$ performs similarly, with the one using $CoronaryCNN_{ROI}$ having the highest mean DSC. The version using $CoronaryCNN_{ROI}$ performs better than the one using $CoronaryCNN_{flip}$ on the pilot 2 volume. This can be seen in figure 4.9k where the version using $CoronaryCNN_{ROI}$ manages to segment more of the coronary artery tree than the version using $CoronaryCNN_{fig}$ (figure 4.9h). The opposite case is true for evaluation on the pilot 3 volume. On this volume, the version using $CoronaryCNN_{flip}$ performs better than the one using $CoronaryCNN_{ROI}$. As shown in figure 4.9i, the version using $CoronaryCNN_{flip}$ is able to extract even more than the coronary tree shown by the ground truth in figure 4.9c, while the version using $CoronaryCNN_{ROI}$ is missing some vessels.

When comparing the three versions some observations can be made. The version using $CoronaryCNN$ fails to segment large parts of the coronary artery, mainly due to the $CoronaryCNN$ producing gaps in the segmentation. Since the proposed method for extracting coronary arteries uses connected component analysis, gaps in the segmentation will lead to discarding responses that are part of the coronary arteries but not connected to the aorta. The version using $CoronaryCNN_{flip}$ manages to segment more of the coro-

nary artery tree, but also has some responses that are not part of the ground truth. Due to being trained on patches that are randomly flipped in all three axises, the network is less likely to overfit on the dataset it was trained on. The version using $CoronaryCNN_{ROI}$ is able to segment large parts of the coronary artery tree and has minimal oversegmentation. $CoronaryCNN_{ROI}$ is also trained on randomly flipped patches, but the samples only come from a small region of interest around the coronary artery. This might lead the network to better learn the boundaries of the coronary arteries.

The results show that there is a large difference in the result of the proposed method from volume to volume. The method is highly sensitive to gaps in the segmentation and responses close to the coronary arteries. The method is not robust, but is in some cases able to give very good results as shown in figure 4.9i. The versions that use a network trained with data augmentation achieves better results. Using a larger training dataset or using more data augmentation to produce more data will most likely improve the networks performance, which will improve the performance of the proposed method. Due to the sensitivity to oversegmentation, the use of a weighting on the surface of the vessels should be investigated as this has been shown to give good results on other domains as discussed by Ronneberger et al. (39).

The results show that it is possible to create a fully automatic segmentation method using deep learning, answering the first research question. The proposed method shows good results on some datasets, but is highly sensitive to gaps in the segmentation which makes the method less robust. The network should be further evaluated on lumen segmentation challenge on the Rotterdam Coronary Artery Evaluation Framework for Stenoses detection and Quantification (22) to obtain more accurate metrics on how the method performs.

5.2 Brain tumor Segmentation

The results of the experiment on brain tumor segmentation is summarized in table 4.4 where the trained network after 15 epochs achieves a validation DSC of 0.8922. The network is able to achieve good segmentation results but is prone to over segmentation as shown in figure 4.10. After 15 epochs the training DSC continues to rise while the validation DSC decreases drastically. This means that the network is starting to learn features that works for the training volumes but not on the validation volumes. Figure 4.11 shows that the trained network is able segment most of the tumors, containing only a small amount of holes and spurious responses.

The figures 4.12b-4.15b show a 3D visualization of the segmentation. In these figures small amounts of spurious results can be observed. As discussed in related work, other methods use a postprocessing step to clean up the spurious responses in segmentation results. Examples of posprocessing steps include using Markov random field (43) or conditional random field (21)

The DeepMedic architecture was originally used for brain tumor and lesion segmentation (21) and the results on the data provided by USIGT (12) shows that the architecture

performs well at problems with large inter-patient variability.

The results show that it is possible to use the same architecture and training parameters as the networks used for aorta and coronary segmentation to train a network on brain tumor segmentation, thus answering the second research question. Due to training on 13 volumes and performing validation on only 4, it is possible that the network is overfitted on the validation data and might not generalize as well to new data. Cross validation needs to be performed to achieve a better DSC metric, by training several networks with different combinations of 13 training and validation volumes and averaging the results.

5.3 Digital rock Segmentation

In this section the results of the digital rock experiments are discussed.

5.3.1 Experiments with different amounts of training data

In the experiments with $RockCNN_1$, $RockCNN_{10}$ and $RockCNN_{40}$ the effects of different amounts of training data was investigated. The training and validation DSC presented in table 4.5 show a clear trend that the performance on the network increases with the amount of training data. This can also be seen from the plots in figures 4.16-4.18. Figure 4.16 shows the training and validation DSC plot on pore segmentation. $RockCNN_1$ shows signs of overfitting, as the training DSC is much higher than the validation DSC. Since $RockCNN_1$ is only trained on one 175^3 subvolume, it is possible that the network manages to learn some special cases that only applies to this volume and does not generalize to the validation volumes. The plot of $RockCNN_{10}$, which has been trained on 10 volumes, show that the validation DSC decreases after 15 epochs, which is likely do to overfitting. Experiment $RockCNN_{40}$ performs better on the validation volumes than on the training volumes. This is due to the samples being selected with equal probability of being centered on a specific class during training. On the validation over the full volumes there is a larger amount of voxels that belong to the pore class, which in turn can increase the validation DSC. The training and validation DSC plots for multi phase (Figure 4.17) shows the same issues. $RockCNN_1$ performs better than $RockCNN_{10}$ and $RockCNN_{40}$ on training, but performs worse on validation. The validation DSC of $RockCNN_{10}$ decreases after 15 epochs for the multi phase segmentation as well. Meanwhile the the training and validation DSC of experiment $RockCNN_{40}$ on multi phase are closer to each other. The same also applies for $RockCNN_1$, $RockCNN_{10}$ and $RockCNN_{40}$ used on grain segmentation, shown in figure 4.18. When the training and validation DSC becomes similar it means that the network is able to generalize and performs more equally on the training and validation subvolumes. Figure 4.22 shows a comparison of a slice of the input, ground truth and the segmentation results. The figure shows that the segmentation results gets progressively more more similar to the ground truth with each step of increased amount of training data.

5.3.2 Experiments with different types of rock

In the experiments with $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$ the effects on training on different types of rock was investigated. The training and validation DSC presented in table 4.5 for $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$ as well as the plots in figures 4.19-4.19 are used to make sure that the networks converges and are not overfitting. The networks are validated on the types of rock they are trained on, which means that direct comparison of the different experiments is not meaningful based on the validation plots alone. Instead the test DSC is shown in tables 4.6-4.9 is used to compare the different networks.

Table 4.6 shows the test DSC for pore, multi phase and grain segmentation on bentheimer sandstone. The network that performs best on this type of rock is $RockCNN_{BT}$, which was trained exclusively on subvolumes of bentheimer sandstone. The second best is $RockCNN_{BT+BR}$, which was trained on equal amounts of bentheimer and berea sandstone. $RockCNN_{BT+BR+CA}$, trained on all three types of rock performs worst on the test DSC across pore, multi phase and grain segmentation.

The test DSC for segmentation on berea sandstone is given by table 4.7. $RockCNN_{BT}$ performs poorly at the berea volumes, a type of rock that it has not been trained on, compared to $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$. It is still able to achieve a test DSC of 0.9789 and 0.9751 for pore and grain segmentation respectively, while DSC for multi phase is 0.4060. This shows that the network is able to learn features from the bentheimer volumes that are useful for berea segmentation. $RockCNN_{BT}$ is outperformed by $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$, both of which has been trained partially on the berea volumes. $RockCNN_{BT+BR+CA}$ produces a test DSC that is slightly worse than $RockCNN_{BT+BR}$, which is similar to the behaviour observed on the test DSC for bentheimer sandstone.

Table 4.8 shows the test DSC for segmentation on the carbonate volumes. Comparing the test DSC of $RockCNN_{BT}$ on berea (Table 4.7) and carbonate shows that the network performs better on pore segmentation in carbonate than on berea volumes. It does however perform much worse for multi phase and grain DSC on carbonate than on berea. This shows that it is much harder for the network to apply the features learned from the bentheimer volumes on the carbonate volumes than on the berea volumes. The test DSC result of $RockCNN_{BT+BR}$ shows a slight increase in performance, but still struggles to produce good segmentations for multi phase and grain. This shows that features learned from a combination of both bentheimer and berea volumes perform better on segmenting carbonate volumes than features learned from bentheimer volumes alone. $RockCNN_{BT}$ and $RockCNN_{BT+BR}$ are both outperformed by $RockCNN_{BT+BR+CA}$, which is trained on all three types of rock.

By taking the average test DSC over the three different types of rock, shown in table 4.9, a trend can be observed. When testing a network on multiple types of rock, the network that has been trained on samples from all types gives the best performance on average. This means that if testing is performed on rock volumes of unknown type it would be

better to use a network that has been trained on a combination of expected rock types. The tables 4.6 and 4.7 show that a network trained on additional types of rocks performs worse than the networks trained only on the type of rock that is being tested. For training the networks there are $2^3 - 1 = 7$ different combinations of types used for training, 3 of which has been investigated in experiments with $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$. Experiments with the remaining 4 permutations should be performed to observe if the best performing networks being the ones trained exclusively on one type of rock is a general result or a special case of the experiments with $RockCNN_{BT}$, $RockCNN_{BT+BR}$ and $RockCNN_{BT+BR+CA}$.

The networks has been tested on different types of rock and the results of this has been discussed. There is however another type of experiments that should be performed. All subvolumes of each type used for training and testing comes from the same larger volumes. By testing on subvolumes sampled from different scans of the same type of rock, it will be possible to investigate how well the networks will perform on new data.

The results show that it is possible to use the same architecture and training parameters as the networks used for aorta and coronary segmentation to train a network on brain tumor segmentation, thus answering the second research question.

5.4 Deep learning framework

In this section the choice of deep learning method is discussed, based on hyperparameter settings, architecture and training time.

5.4.1 Hyperparameters

The choice of hyperparameters for training are highly dependent on the chosen architecture. In this thesis the hyperparameters for both the architecture and training has not been explored other than some data augmentation. Finding an optimal combination of hyperparameters is a challenging task and requires large amount of testing (14). In this thesis a broader approach has been chosen, to use the same configuration and observe the effects on several different types of data. The DeepMedic framework (21) allows for testing different sets of hyperparameters in an easy manner, meaning that it is possible to go deeper into the details of tuning the hyperparameters for one of the segmentation tasks. This will require large amount of time, as one test of a set of hyperparameters can take as long as 8 hours on a modern GPU. By utilizing several machines with GPUs it is possible to parallelize the hyperparameter search.

As discussed, the network has problems with overfitting when working on small amounts of data. One of the reason behind this could be that the network has too many weights, giving too many degrees of freedom (14). The trained networks that overfit usually have a high training DSC, which means that it is likely has enough weights to map the input samples to the corresponding output segmentations internally. Experiments lowering the amount of weights could be performed. The fully connected layers use dropout for regu-

larization. This can be increased to try to force the network to generalize, but will most likely lead to slower convergence of the network (14). The most straightforward way to limit overfitting is to increase the amount of training data. If proper data is provided, this will give the network a larger variance in training samples. This is however not always possible, such as limited amount of training data for certain medical image sets as discussed by Litjens et al. (27). Another way to increase the amount of data is to use data augmentation.

5.4.2 Data augmentation

As shown in the results for coronary artery segmentation, data augmentation has a positive impact on decreasing the amount of overfitting. This leads to the networks segmenting more of the coronary arteries, but at the same time more spurious responses, with the net effect being a lower DSC compared to not using data augmentation. Several methods discussed in related work uses only a small amount of datasets, such as Cicek et al. (6) which trained on only two volumes and managed to generalize well on a third volume. Both Cicek et al. (6) and Milletari et al. (32) use a large amount of data augmentation. This is performed by deforming the input volumes randomly with elastic deformation each optimization step. The same concept could be tried on the coronary artery segmentation task to generate more data.

5.4.3 Other architectures

In this thesis a single architecture has been chosen and used for all segmentation experiments. The architecture works on small image segments and uses two pathways to increase the spatial area that the network is able to use as information. From the results it can be seen that the chosen architecture performs much better on aorta segmentation, brain tumor segmentation and digital rock segmentation, compared to coronary artery segmentation. For the brain tumor segmentation task the tumors have a large variance in both shape and position and does not require spatial information relative to the brain. The same concept applies to the digital rock segmentation task, where information about the position relative to the whole image might be less relevant. The results for aorta segmentation shows that the network performs much better on volumes resampled to 0.80mm voxel spacing than on volumes resampled to 0.40mm. A reason for this could be that the network trained on 0.40mm voxel spacing does not receive enough spatial information to determine if the voxels should be classified as part of the aorta or not. For coronary artery segmentation the experiment with resampling to 0.80mm voxel spacing does not improve the DSC significantly. A reason for this is that the coronary arteries are already represented by a small amount of voxels in the 0.40mm voxel spacing volumes, leading to loss of information when resampling to 0.80mm. Compared to the other segmentation tasks, the aorta and coronary arteries will almost always be in the same regions in the image relative to the heart. Several methods discussed in related work uses statistical models to guide the search for coronary arteries (56). Other architectures working on whole images (6; 32) can be tried for coronary artery extraction to see if the architectures are able to learn where the vessels are in relation to the heart. There are however issues with training these types of

networks, as they all require large FMs in early and late layers to be able to work on the complete image.

5.4.4 Training time

Table 4.10 shows the training and inference times for the networks used in this thesis. The table shows that the training times are divided into two groups, one using 19 to 20 hours and the other using approximately 8 hours. The common factor of the networks using 19 to 20 hours is that they work on larger volumes due to being resampled to 0.40mm. The mean size is approximately 500x500x300. The size of the volumes resampled to 0.80mm voxel spacing are 8 times smaller, but still has approximately 8 hours training time. The volumes used for brain tumor segmentation are approximately 200x200x150, while the rock subvolumes are 175x175x175, both of which gives approximately 8 hours training time.

During training of the network, all networks will be trained on the same number of equally sized 3D patches. This should mean that the training time should be approximately the same for all datasets. However for the larger volumes, a large amount of time is spent on reading the volumes from the disk and extracting patches. For the smaller volumes the software manages to extract the patches for the next subepoch in parallel while training and validating. The volumes used for coronary artery segmentation can not be split into subvolumes, as this might introduce class imbalance between the subvolumes. Large speedups for training can be made by caching a large amount of image patches instead of extracting them for each subepoch.

The table 4.10 shows that the inference time for the different networks mainly depends on the size of the input volume. For the coronary artery segmentation task, the inference times could be sped up by only performing inference inside a region of interest of the heart. This does however require a method to first segment the heart, similarly to the aorta segmentation network.

Chapter 6

Conclusion and future work

6.1 Conclusion

In this thesis a method for fully coronary artery segmentation using deep learning has been proposed, implemented and evaluated on a dataset that has been manually segmented by clinical experts at St. Olavs Hospital. The results of the method varies from being able to extract the full coronary tree on some images, while missing some branches in others. The method uses two networks trained on aorta and coronary artery segmentation respectively. The result of the proposed method is highly depending of the quality of the segmentations produced by these networks. It is not robust against small gaps in the segmentation, which lead to parts of the coronary artery tree being mistakenly discarded. The results are however promising seeing that a network trained on only 6 volumes is able to generalize on a new dataset. This answers the first research question of this thesis.

The second research question asked if it was possible to use the same type of deep learning method to perform fully automatic segmentation of brain tumors from MRI volumes? In this thesis the same architecture and training parameters as the ones used for aorta and coronary artery segmentation have been used to train a network to perform fully automatic segmentation of brain tumors. The dataset that was used for training and validation of the network was provided by USIGT (12). The results show that the network is able to achieve high validation scores. The method does however produce a substantial amount of spurious responses for some of the MRI volumes. The method shows promising results and has potential to obtain better results by providing more training data and fine-tuning the hyperparameters of the network, thus answering the second research question.

The last research question asked if the same deep learning method can be trained perform digital rock segmentation. In this thesis the same architecture and training parameters as the ones used for aorta and coronary artery segmentation have been used to train a network to perform fully automatic segmentation of digital rock. The dataset used for training and testing was provided by FEI (1). In this thesis the chosen method is trained and tested on

the dataset and produced good results, both quantitatively and qualitatively. Experiments with different amount of data shows that the deep learning method performs better and is less likely to overfit with larger amounts of training data.

6.2 Future work

In this section the future work is discussed. The future work for coronary artery segmentation is first presented, followed by future work on brain tumor segmentation and digital rock segmentation. Lastly future work for the deep learning framework used in this thesis is discussed.

6.2.1 Coronary Artery Segmentation

The proposed method is able to segment large parts of the coronary tree but is not robust against small gaps in the segmentation. Ideally the network segmenting the coronary arteries should not produce gaps. To improve the performance of the network for coronary artery segmentation the best solution would be to train it on more data. This makes it less likely for the trained network to overfit on a small sample of training volumes and will help generalize on new datasets. As discussed in the discussion chapter, training the network on a small region of interest around the coronary arteries has the effect of producing a network that segments a large amount of tubular structures. This is shown to work better in combination with the proposed method than the networks trained on a larger amount of negative samples during training. The results also show that data augmentation has a positive impact on the segmentation result. In this thesis the only form of data augmentation was randomly flipping the training patches along all three axises. More sophisticated data augmentation methods, such as random elastic deformation should be tested on the coronary artery segmentation task. Since the method is sensitive to oversegmentation, the use of a weighting on the vessel boundary during training should be explored.

The pilot dataset used to evaluate the proposed method in this thesis does not provide an accurate evaluation, mostly due to the aorta and coronary arteries ground truth being one model. Proper evaluation of the proposed method should be performed by using the lumen segmentation challenge on the Rotterdam Coronary Artery Algorithm Evaluation Framework for Stenoses Detection and Quantification (22).

6.2.2 Brain tumor segmentation

The results of the network trained to perform fully automatic segmentation of brain tumors are promising. The segmentation output manages to achieve a high DSC score, however the network shows signs of overfitting. Fine-tuning the hyperparameters of the network and training or using data augmentation could decrease the chance of overfitting and improve the segmentation result. Another issue is the segmentation of spurious responses. Methods for refining the output segmentation should be looked into, similarly to the method used by Kamnitsas et al. (21). The experiments presented in this thesis

for brain tumor segmentation should be cross validated by training the network multiple times, but with different combinations of training and validation volumes.

6.2.3 Digital rock segmentation

The results of the networks trained to perform fully automatic segmentation on digital rocks show good performance of the networks. The amount of subvolumes used for training the networks should be increased until an increase in validation DSC can no longer be observed, to find how many subvolumes are needed to obtain good results. Further fine-tuning of the architecture and training hyperparameters should also be tested. When training networks on different types of rocks there are several combinations that were not tested. These permutations should be tested to see if there are some types rock segmentation that benefits from being trained on multiple types. Another test that should be performed is to evaluate a network trained on subvolumes from one scan on subvolumes from a different scan of the same type of rock.

6.2.4 Deep learning framework

The timing results show that the software framework used in this thesis (21) has long training times for large volumes. It is not possible to split the coronary artery dataset volumes into smaller subvolumes as this will introduce class imbalance. Instead of extracting the training patches each sub epoch, it would be possible to cache a larger amount of patches to reduce the I/O time of the framework. Modifying the framework to handle larger volumes should be investigated. If the training time on large volumes can be reduced from 20 to 8 hours, it means that it will be possible to train more versions of the networks in the same amount of time, making it easier to fine-tune the architecture and training hyperparameters.

In thesis the deep learning architecture uses small patches of the volume as input. Other architectures that are able to use information the whole volume at once such as the methods proposed by Cicek et al. (6) and Milletari et al. (32) should be explored.

Bibliography

[1] Discover digital rock, Jul 2016. Accessed Jun 1, 2017. https://www.fei.com/oil-gas/products-services/digital-rock/.

[2] ANTIGA, L., PICCINELLI, M., BOTTI, L., ENE-IORDACHE, B., REMUZZI, A., AND STEINMAN, D. A. An image-based modeling framework for patient-specific computational hemodynamics. *Medical & biological engineering & computing 46*, 11 (2008), 1097.

[3] BARSNESS, G. W., AND HOLMES, D. R. *Coronary Artery Disease: New Approaches Without Traditional Revascularization*. Springer Science & Business Media, 2011.

[4] BAUER, C., AND BISCHOF, H. Edge based tube detection for coronary artery centerline extraction. *The Insight Journal* (2008).

[5] BLEEKER, F. E., MOLENAAR, R. J., AND LEENSTRA, S. Recent advances in the molecular understanding of glioblastoma. *Journal of neuro-oncology 108*, 1 (2012), 11–27.

[6] ÇIÇEK, Ö., ABDULKADIR, A., LIENKAMP, S. S., BROX, T., AND RONNEBERGER, O. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2016), Springer, pp. 424–432.

[7] DEHKORDI, M. T., SADRI, S., AND DOOSTHOSEINI, A. A review of coronary vessel segmentation algorithms. *Journal of medical signals and sensors 1*, 1 (2011), 49.

[8] DICE, L. R. Measures of the amount of ecologic association between species. *Ecology 26*, 3 (1945), 297–302.

[9] FILIPOIU, F. M. *Atlas of heart anatomy and development*. Springer Science & Business Media, 2013.

[10] FRANGI, A. F., NIESSEN, W. J., VINCKEN, K. L., AND VIERGEVER, M. A. Multiscale vessel enhancement filtering. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (1998), Springer, pp. 130–137.

[11] FRIMAN, O., KÜHNEL, C., AND PEITGEN, H.-O. Coronary centerline extraction using multiple hypothesis tracking and minimal paths. In *Proc MICCAI* (2008), vol. 42.

[12] FYLLINGEN, E. H., STENSJØEN, A. L., BERNTSEN, E. M., SOLHEIM, O., AND REINERTSEN, I. Glioblastoma segmentation: Comparison of three different software packages. *PloS one 11*, 10 (2016), e0164891.

[13] GALLEGO, O. Nonsurgical treatment of recurrent glioblastoma. *Current oncology 22*, 4 (2015), e273.

[14] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[15] GÜLSÜN, M. A., FUNKA-LEA, G., SHARMA, P., RAPAKA, S., AND ZHENG, Y. Coronary centerline extraction via optimal flow paths and cnn path pruning. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2016), Springer, pp. 317–325.

[16] HAVAEI, M., GUIZARD, N., CHAPADOS, N., AND BENGIO, Y. Hemis: Hetero-modal image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2016), Springer, pp. 469–477.

[17] HE, K., ZHANG, X., REN, S., AND SUN, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 1026–1034.

[18] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[19] IOFFE, S., AND SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[20] JACOBS, M. A., IBRAHIM, T. S., AND OUWERKERK, R. Mr imaging: Brief overview and emerging applications 1. *Radiographics 27*, 4 (2007), 1213–1229.

[21] KAMNITSAS, K., FERRANTE, E., PARISOT, S., LEDIG, C., NORI, A., CRIMINISI, A., RUECKERT, D., AND GLOCKER, B. Deepmedic on brain tumor segmentation. *Athens, Greece Proc. BRATS-MICCAI* (2016).

[22] KIRIŞLI, H., SCHAAP, M., METZ, C., DHARAMPAL, A., MEIJBOOM, W. B., PAPADOPOULOU, S.-L., DEDIC, A., NIEMAN, K., DE GRAAF, M., MEIJS, M., ET AL. Standardized evaluation framework for evaluating coronary artery stenosis detection, stenosis quantification and lumen segmentation algorithms in computed tomography angiography. *Medical image analysis 17*, 8 (2013), 859–876.

[23] KITAMURA, Y., LI, Y., ITO, W., AND ISHIKAWA, H. Coronary lumen and plaque segmentation from cta using higher-order shape prior. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2014), Springer, pp. 339–347.

[24] KITSLAAR, P. H., FRENAY, M., OOST, E., DIJKSTRA, J., STOEL, B., AND REIBER, J. Connected component and morpholgy based extraction of arterial centerlines of the heart (cocomobeach). *The Midas Journal* (2008).

[25] LESAGE, D., ANGELINI, E. D., BLOCH, I., AND FUNKA-LEA, G. A review of 3d vessel lumen segmentation techniques: Models, features and extraction schemes. *Medical image analysis 13*, 6 (2009), 819–845.

[26] LI, K., WU, X., CHEN, D. Z., AND SONKA, M. Optimal surface segmentation in volumetric images-a graph-theoretic approach. *IEEE transactions on pattern analysis and machine intelligence 28*, 1 (2006), 119–134.

[27] LITJENS, G., KOOI, T., BEJNORDI, B. E., SETIO, A. A. A., CIOMPI, F., GHAFOORIAN, M., VAN DER LAAK, J. A., VAN GINNEKEN, B., AND SÁNCHEZ, C. I. A survey on deep learning in medical image analysis. *arXiv preprint arXiv:1702.05747* (2017).

[28] LONG, J., SHELHAMER, E., AND DARRELL, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3431–3440.

[29] LUGAUER, F., ZHANG, J., ZHENG, Y., HORNEGGER, J., AND KELM, B. M. Improving accuracy in coronary lumen segmentation via explicit calcium exclusion, learning-based ray detection and surface optimization. In *SPIE Medical Imaging* (2014), International Society for Optics and Photonics, pp. 90343U–90343U.

[30] LUGAUER, F., ZHENG, Y., HORNEGGER, J., AND KELM, B. M. Precise lumen segmentation in coronary computed tomography angiography. In *International MICCAI Workshop on Medical Computer Vision* (2014), Springer, pp. 137–147.

[31] MAIER, O., MENZE, B. H., VON DER GABLENTZ, J., HÄNI, L., HEINRICH, M. P., LIEBRAND, M., WINZECK, S., BASIT, A., BENTLEY, P., CHEN, L., ET AL. Isles 2015-a public evaluation benchmark for ischemic stroke lesion segmentation from multispectral mri. *Medical image analysis 35* (2017), 250–269.

[32] MILLETARI, F., NAVAB, N., AND AHMADI, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3D Vision (3DV), 2016 Fourth International Conference on* (2016), IEEE, pp. 565–571.

[33] MOESKOPS, P., WOLTERINK, J. M., VAN DER VELDEN, B. H., GILHUIJS, K. G., LEINER, T., VIERGEVER, M. A., AND IŠGUM, I. Deep learning for multi-task medical image segmentation in multiple modalities. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2016), Springer, pp. 478–486.

[34] MOHR, B., MASOOD, S., AND PLAKAS, C. Accurate lumen segmentation and stenosis detection and quantification in coronary cta. In *Proceedings of 3D Cardiovascular Imaging: a MICCAI segmentation challenge workshop* (2012).

[35] NABEL, E. G., AND BRAUNWALD, E. A tale of coronary artery disease and myocardial infarction. *New England Journal of Medicine 366*, 1 (2012), 54–63.

[36] NICHOLS, M., TOWNSEND, N., SCARBOROUGH, P., AND RAYNER, M. Cardiovascular disease in europe: epidemiological update. *European heart journal 34*, 39 (2013), 3028–3034.

[37] PEREIRA, S., PINTO, A., ALVES, V., AND SILVA, C. A. Brain tumor segmentation using convolutional neural networks in mri images. *IEEE transactions on medical imaging 35*, 5 (2016), 1240–1251.

[38] PIEPER, S., HALLE, M., AND KIKINIS, R. 3d slicer. In *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on* (2004), IEEE, pp. 632–635.

[39] RONNEBERGER, O., FISCHER, P., AND BROX, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2015), Springer, pp. 234–241.

[40] RUBIN, G. D., LEIPSIC, J., SCHOEPF, U. J., FLEISCHMANN, D., AND NAPEL, S. Ct angiography after 20 years: a transformation in cardiovascular disease characterization continues to advance. *Radiology 271*, 3 (2014), 633–652.

[41] SCHAAP, M., METZ, C. T., VAN WALSUM, T., VAN DER GIESSEN, A. G., WEUSTINK, A. C., MOLLET, N. R., BAUER, C., BOGUNOVIĆ, H., CASTRO, C., DENG, X., ET AL. Standardized evaluation methodology and reference database for evaluating coronary artery centerline extraction algorithms. *Medical image analysis 13*, 5 (2009), 701–714.

[42] SETIO, A. A. A., CIOMPI, F., LITJENS, G., GERKE, P., JACOBS, C., VAN RIEL, S. J., WILLE, M. M. W., NAQIBULLAH, M., SÁNCHEZ, C. I., AND VAN GINNEKEN, B. Pulmonary nodule detection in ct images: false positive reduction using multi-view convolutional networks. *IEEE transactions on medical imaging 35*, 5 (2016), 1160–1169.

[43] SHAKERI, M., TSOGKAS, S., FERRANTE, E., LIPPE, S., KADOURY, S., PARAGIOS, N., AND KOKKINOS, I. Sub-cortical brain structure segmentation using f-cnn's. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on* (2016), IEEE, pp. 269–272.

[44] SMISTAD, E., ELSTER, A. C., AND LINDSETH, F. Gpu accelerated segmentation and centerline extraction of tubular structures from medical images. *International journal of computer assisted radiology and surgery 9*, 4 (2014), 561–575.

[45] SMISTAD, E., FALCH, T. L., BOZORGI, M., ELSTER, A. C., AND LINDSETH, F. Medical image segmentation on gpus–a comprehensive review. *Medical image analysis 20*, 1 (2015), 1–18.

[46] SUTSKEVER, I., MARTENS, J., DAHL, G. E., AND HINTON, G. E. On the importance of initialization and momentum in deep learning. *ICML (3) 28* (2013), 1139–1147.

[47] TEK, H., GULSUN, M. A., LAGUITTON, S., GRADY, L., LESAGE, D., AND FUNKA-LEA, G. Automatic coronary tree modeling. *The Insight Journal* (2008).

[48] THEANO DEVELOPMENT TEAM. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints abs/1605.02688* (May 2016).

[49] TIELEMAN, T., AND HINTON, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning 4*, 2 (2012).

[50] WANG, C., MORENO, R., AND SMEDBY, Ö. Vessel segmentation using implicit model-guided level sets. In *MICCAI Workshop" 3D Cardiovascular Imaging: a MICCAI segmentation Challenge", Nice France, 1st of October 2012.* (2012).

[51] XU, C., AND PRINCE, J. L. Gradient vector flow: A new external force for snakes. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on* (1997), IEEE, pp. 66–71.

[52] YANG, G., KITSLAAR, P., FRENAY, M., BROERSEN, A., BOOGERS, M. J., BAX, J. J., REIBER, J. H., AND DIJKSTRA, J. Automatic centerline extraction of coronary arteries in coronary computed tomographic angiography. *The international journal of cardiovascular imaging 28*, 4 (2012), 921–933.

[53] YOUNG, R. M., JAMSHIDI, A., DAVIS, G., AND SHERMAN, J. H. Current trends in the surgical management and treatment of adult glioblastoma. *Annals of translational medicine 3*, 9 (2015).

[54] ZAMBAL, S., HLADUVKA, J., KANITSAR, A., AND BÜHLER, K. Shape and appearance models for automatic coronary artery tracking. *The Insight Journal 4* (2008).

[55] ZHAO, L., AND JIA, K. Multiscale cnns for brain tumor segmentation and diagnosis. *Computational and mathematical methods in medicine 2016* (2016).

[56] ZHENG, Y., TEK, H., AND FUNKA-LEA, G. Robust and accurate coronary artery centerline extraction in cta by combining model-driven and data-driven approaches. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2013), Springer, pp. 74–81.