Data Augmentation to Increase Multi-Site Robustness for Convolutional Neural Networks

A case study on MRI segmentation of target and organs at risk for prostate cancer

Frida Börnfors, Elisabeth Klint

Advisors: Stefan Adalbjörnsson, Andreas Jakobsson
Master Thesis at the Faculty of Mathematical Statistics, LTH

June 2019

LUND
UNIVERSITY

Abstract

Organ segmentation on magnetic resonance (MR) images for dose planning on cancer patients is a time consuming process that can be automatized by using convolutional neural networks (CNNs). MR images vary greatly in characteristics across acquisition sites, which affects CNN segmentation performance. In this master thesis, augmentation methods were tested and implemented to increase the robustness of CNNs in a multi-site context.

A data set of MR images from 151 prostate cancer patients from four Swedish hospitals (denoted A - D) was used, and the prostate, bladder, rectum, femoral heads, and body were segmented. Four networks were developed on 40 training subjects from a single site for two vendors using $HighRes3DNet\_large$ implemented in NiftyNet. Two networks were trained without augmentation, and two with. The networks were evaluated on average Dice score (DSC) over all organs, on a validation set consisting of images from all four sites. The DSC improved from 0.886 to 0.927 for the network trained on site A, and from 0.668 to 0.919 for the network trained on site C when augmentation was added.

The applied augmentations were either a part of the NiftyNet framework, namely bias field and geometric augmentations including elastic deformation, or implemented by the authors, i.e., augmentations by histogram modification, adding noise, and smoothing. All augmentations were applied online with the aim to increase the variety during training of the CNN. Hence, the augmentations were not intended to mimic the non-training sites.

The DSCs when using augmentation are comparable to scores for a network trained on a large multi-site data set (0.930) as well as organ segmentation variability between experts. The DSC is based on pixel overlap between the network segmentation and the ground truth, which was segmented by non-experts using available guidelines. This thesis shows the importance of augmentation for multi-site segmentation and presents useful tools as a stepping stone towards automatizing organ segmentations.

Acknowledgements

We would like to express our appreciation towards the team at Spectronic Medical for all the insightful ideas and support, as well as the opportunity to conduct this thesis with them. A special thanks to our supervisor Stefan Adalbjörnsson for his sharing of knowledge, tireless guidance when discussing our work, and help with solving our mini-crises.

A warm thanks to the team at the Department of Hematology, Oncology, and Radiation Physics at Skånes Universitetssjukhus (SUS) in Lund, especially Emilia Persson and Christian Jamtheim Gustafsson, whose time and expertise were immensely valuable for understanding the clinical aspects concerning this thesis.

Further, we want to acknowledge the valuable feedback on presentations and report, as well as reading advice, given by our supervisor at LTH, Andreas Jakobsson. The feedback from our examiner, Johan Swärd, during the final stages of this thesis was also greatly appreciated.

We would like to express our gratitude to the Gentle Radiotherapy consortium for allowing access to the magnetic resonance imaging (MRI) data used in this work, constituting a subset of the data acquired during the MR-OPERA study by Persson et al. [1]. All MRI data was managed with the highest level of confidentiality and security throughout the entire project. All MR images marked "courtesy of Gentle Radiotherapy" reproduced in this report are approved by Gentle Radiotherapy. Furthermore, we are thankful for the computational resources provided by *Analytic Imaging Diagnostics Arena, AIDA* [2].

Last but not least, we would like to thank our families and friends for all the time and energy spent listening, reading, and discussing every bit of this thesis.

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial intelligence
ANN	Artificial neural network
CNN	Convolutional neural network
CT	Computed tomography
CTV	Clinical target volume
DL	Deep learning
DNN	Deep neural network
DSC	Dice score
FOV	Field of view
GPU	Graphical processing unit
ML	Machine learning
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
OAR	Organs at risk
PD	Proton density
PDF	Probability density function
PTV	Planning target volume
ReLU	Rectified linear unit
RT	Radiotherapy, Radiation therapy
sCT	Synthetic CT
SGD	Stochastic gradient descent
SNR	Signal-to-noise ratio
SUS	Skåne University Hospital

Chapter 1

Introduction

1.1 Background

In 2016, prostate cancer was the most common type of cancer in Sweden, representing 16.3% of all cancer cases with almost 10 500 men diagnosed with the disease during that year [3]. There are different treatment options for prostate cancer, and the Swedish national guidelines recommends treatment type based on several factors, for example tumor propagation and patient health [4]. During 2016, nearly 2 300 men received external radiotherapy (RT) as a primary or secondary treatment, either together with brachytherapy or RT alone [5].

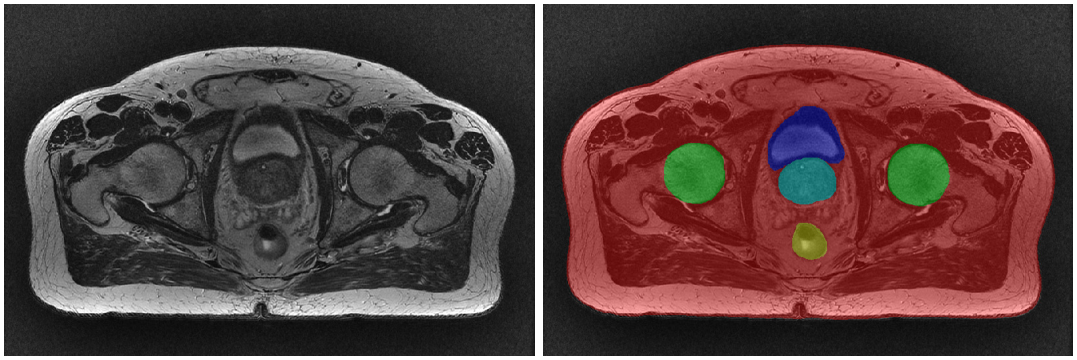
When a patient is planned to receive external radiation therapy, a Computed Tomography (CT) and/or a Magnetic Resonance (MR) image is taken for the dose planning procedure. The treatment plan is based on the intensities in the CT image as well as placement and size of the prostate and surrounding organs, also referred to target and organs at risk (OARs). A dose matrix determines how much radiation the patient is given from each angle and is optimized to target the prostate while reducing the risk of harming any OARs.

To calculate how much radiation each organ receives during treatment, a delineation, also referred to as segmentation or annotation, of the organ is required. The delineation can be made using the CT image, the MR image, or both, where the different modalities contrast different features. The MR images are superior in soft-tissue contrast, easing the delineation procedure. Figure 1.1 shows an example of an MR image slice and the corresponding segmentations of body, femoral heads, bladder, rectum, and prostate.

The organ delineation process is usually manual or semi-automatic and is a time consuming process. Furthermore, an inter-expert variability in the delineation process is present, as explained in Section 2.4.2. Automatic ways of segmenting the different organs is thus of interest as it can ease the workload of clinical experts and has the potential to reduce the delineation variability.

One way of approaching this problem is by using convolutional neural networks (CNNs), which often achieves state-of-the-art performance [6, 7]. Given an image as input, a CNN can output a suggested segmentation. A CNN learns how to perform the delineations by learning from a training data set consisting of MR images and the corresponding segmentations, where the latter are referred to as ground truth. The key concept is for the network to learn features across the training set that generalize to unseen data. The network features can then be applied on new data without ground truth to produce a segmentation. In the case of this thesis, a data set of MR images from four sites has been used in order to develop methods to increase robustness for CNN segmentation. An example of an MR image with an overlaid ground truth segmentation can be seen in Figure 1.1b.

One key issue with network segmentations on MR images is that the images can vary greatly in characteristics and quality depending on acquisition site. For instance, the variation could stem from vendor and camera settings. A neural network that is not trained on images from



(a) An MR slice

(b) An MR slice with organ segmentations; femoral heads (green), bladder (blue), prostate (turquoise), and rectum (yellow). The red segment outlines the rest of the body.

Figure 1.1: An example of an MR slice[1] *with corresponding segmentation.*

multiple sites, will most likely not be able to infer knowledge from training onto images from new sites. Providing more examples or including more sites in the training set exposes the network to a wider range of images which often results in better performance. However, this is not always possible as medical images with expert segmentations are a scarcity. This problem and potential solutions are of interest not only within the prostate cancer field. Other ways to increase the variation in the training set than collection of more images are of interest. A common approach is to modify images in the training set in various ways and to different extents to expose the network for "new" images. These modifications are also called data augmentation.

1.2 Aim

The aim of this thesis is to investigate if data augmentation can improve the robustness for multi-site segmentation using a CNN. In other words, improve performance where the distribution of the training and validation sets differ. This was done in collaboration with Spectronic Medical AB and using an MR image data set provided by *Gentle Radiotherapy* [8].

To achieve this, the following questions have been addressed in this report:

- If training a neural network on data from a single site with and without augmentation
  - Will the network perform better on that site than on other sites, as stated in theory?
  - Will the network perform better on sites with the same camera vendor as the training site than on sites with a different vendor?
  - Is it possible to achieve the same performance on all sites?
  - Can the same result be achieved as when using a multi-site data set?
- What impact does the choice of training site have on the questions posed above?

[1] Original MR image courtesy of Gentle Radiotherapy

1.3 Agenda

In this thesis, relevant theory behind CNNs, Magnetic Resonance Imaging (MRI) and multi-site issues is given in Chapter 2. It is followed by an overview of the data and its characteristics, as well as the used software and hardware (Chapter 3). The methodology and approach to answer the previously posed questions are presented in Chapter 4. Then, the acquired results are presented (Chapter 5) and discussed together with general implications and suggestions for further research (Chapter 6). The main findings are concluded in Chapter 7.

Chapter 2

Theory

This chapter provides the reader with relevant theory within the area of machine learning (ML), and more specifically, CNNs. Theory behind magnetic resonance imaging and the radiotherapy (RT) planning workflow is also covered which introduces important topics for this thesis, such as multi-site problems and manual organ segmentation.

2.1 Artificial intelligence

The areas within artificial intelligence (AI) are many and possibly confusing. The following section aims to clarify some of them with basis in the conceptual illustration in Figure 2.1.

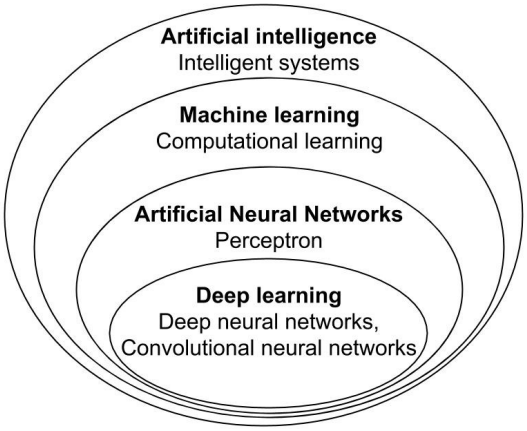


Figure 2.1: Relation between AI, ML, ANN, and DL.

AI is a unifying concept for systems developed to display some form of intelligence or similarity to the human brain which is commonly modeled by algorithms [9, 10]. One sub concept of AI using statistical modeling on the data is ML. Data examples are used to identify traits that for example can be used to predict or identify new data, meaning that traits are identified without explicit lines of code to extract them [9]. More specifically, considering recognizing

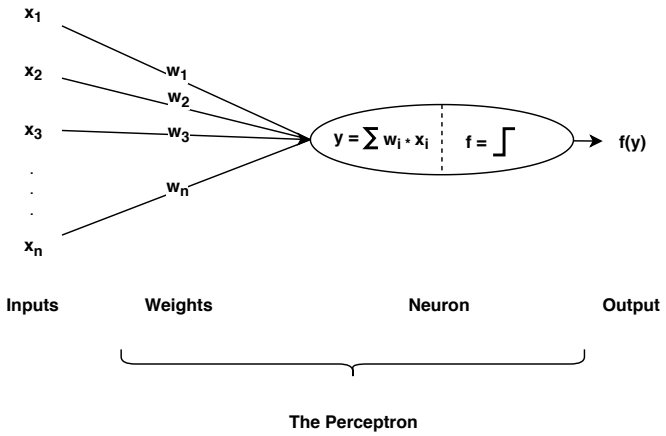


Figure 2.2: The flow from input to output for a single layer perceptron. The activation function is in this case a Heaviside step function.

patterns, classifying or predicting, artificial neural networks (ANN) are widely used for image recognition due to their adaptability and mapping from input to output [11].

To handle larger data sets using a more complex network structure, deep learning (DL), and learning by deep neural networks (DNNs) were introduced as a further development of ML [9, 11]. Convolutional Neural Network (CNN) is a type of deep neural network that is used for many image segmentation applications [11, 12].

There is a distinction between supervised and unsupervised methods in ML. Supervised learning means that the desired output is known, e.g., given an image as input, the AI system should learn to segment organs by using the known ground truth. Unsupervised learning on the other hand, uses similarity traits to correctly place previously unseen data into categories mapped out from the training data [11].

2.1.1 Artificial neural networks

An ANN is a computational system which can be used for a variety of applications. It has resemblance with biological neural circuits which have neurons accepting inputs from many sources and producing an output, if activated.

The simplest model of a neural network is a single layer perceptron, which is illustrated in Figure 2.2. It accepts n inputs $x = [x_1, x_2, \ldots, x_n]^T$ which are multiplied with the weights $w = [w_1, w_2, \ldots, w_n]^T$. The neuron, often referred to as a node in the artificial case, sums the weighted inputs as

$$y = \sum_{i=1}^{n} w_i x_i = \mathbf{w}^T \mathbf{x}, \tag{2.1}$$

and uses an activation function, f, to produce the output, $f(y)$ [13, 14]. In a simple case, the input could be patient information (temperature, blood pressure, weight, age) that is weighted and summed. The activation function can be a step function which compares the sum to a threshold, where the output can be selected to be 0 or 1 describing whether the patient is sick or not. It is common to add a bias term, b, that alters at what value the activation function is

triggered, and the equation for the artificial neuron is modified to

$$y = \sum_{i=1}^{n} w_i x_i + b = \mathbf{w}^T \mathbf{x} + b. \tag{2.2}$$

The system can be up-scaled, as more nodes are added to the structure, either in parallel (several nodes in the same layer) or in series (several layers of nodes).

The underlying idea of ANNs is to find patterns or features in the input and produce a desired output based on the identified properties. This is done by training the network, i.e., learning the weights which produce the best result according to a specified function, termed loss or cost function. This is an iterative process where the network takes a batch of training examples, applies the network operations (multiplies with weights, summing and putting through the activation functions) and calculates the loss. Based on how the network performed, the weights will be adjusted, and the process starts over. There are many options on how to train a network, some of which are explained in Section 2.2.1. After the training process, inference can take place, i.e., inferring knowledge by applying learned weights on data without knowing the ground truth.

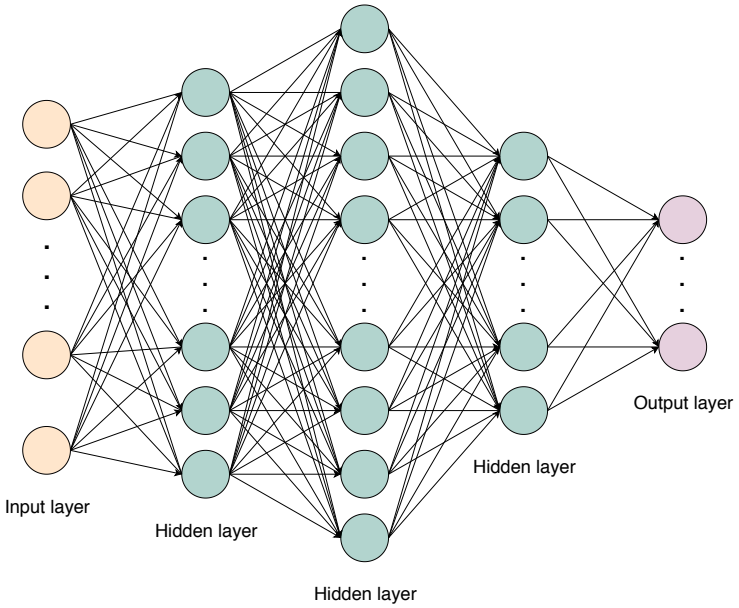


Figure 2.3: An example of a deep neural network with three hidden layers and a various number of nodes in each layer.

2.1.2 Deep neural networks

A deep neural network follows the same principle as the single layer perceptron described in Section 2.1.1, but has multiple hidden layers of nodes, located between the input and the output layer, see Figure 2.3. Goodfellow, Bengio, and Courville [15] explain that multiple layers provide the possibility to create a more complex AI system since simpler concepts can be combined within the network to a more complex concept. For an image, the simple concepts could be edges and corners that are combined into a specific organ, which represents a more complex concept. Mathematically, multiple layers allows representation of more complex functions.

2.1.3 Convolutional neural networks

The number of weights to be learned by a network quickly escalates as the number of inputs increases. For example, an image with 512×512 pixels contributes with $512 \cdot 512 = 262\ 144$ pixel values as inputs, meaning that as many weights need to be learned for the single layer perceptron only. A deep neural network as described above would have a very large amount of weights to be learned. To overcome this and be able to work with images, a special type of deep neural network is used, namely the convolutional neural network.

Instead of using one weight per pixel, the network consists of several sliding convolutional filters. A sliding convolutional filter, or sliding kernel, is a matrix of weights that slides over the input and produces an output value for each position of the filter. During training, the network learns which weights are needed in order to find useful features in the images. Since the filters slide across the image, each filter attempts to identify features that are useful across the whole image, regardless of where the filter is applied. The filters are usually rather small, for example of size 3×3.

A convolution weighs and adds a number of elements together, as specified by the filter. Applying convolution to an image is defined by

$$y[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m,n]x[i-m, j-n], \tag{2.3}$$

where $y[i,j]$ is the output pixel after applying the filter h on the input pixel $x[i,j]$. In the case of a 3×3 convolutional filter, (2.3) can be simplified to

$$y[i,j] = \sum_{m=-1}^{1} \sum_{n=-1}^{1} h[m,n]x[i-m, j-n]. \tag{2.4}$$

In order not to lose information in the corners of the image and keep the size of the original image from input to output, a padding can be added. Figure 2.4a shows a more illustrative way of how convolution in image processing works.

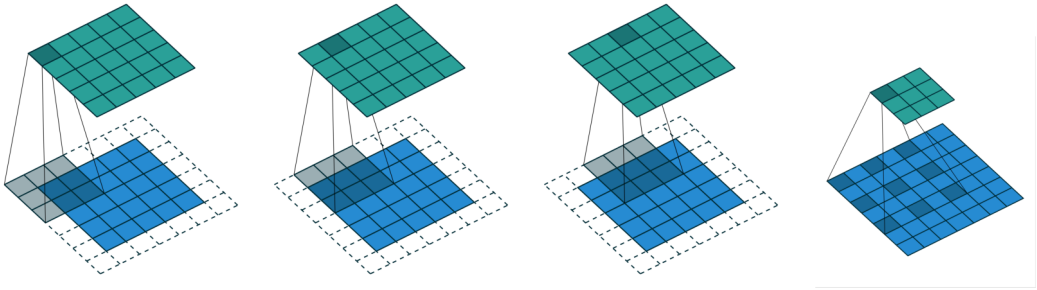
A dilated convolution is similar to the convolution previously described, but when choosing pixels to convolve with the filter, a defined gap is added [16]. As illustrated in Figure 2.4b, using a dilation of two will skip one pixel between each pixel that is used for the convolution operation. A dilation of one is the same as the normal convolution operation.

A CNN consists of several blocks, each containing a number of convolutional filters and an activation function. For the filters within a block, the output of one filter is the input of the next. The architecture of the network specifies the number of filters within each block, size of filters, types of convolution, which activation functions are used, etc.

2.2 Network parameters

As the overall structure of neural networks have been previously explained, this section will cover practical matters which are important when implementing a neural network, regardless if it is a CNN or another type of ANN.

The available data is typically split into training, validation, and test set which are used for different purposes. The training set is fed through the network to learn suitable weights. The validation set is used to evaluate the network's performance throughout the training process as a tool for the user to measure how well the network can generalize to data it was not trained on. The performance on the validation set helps the user to tune parameters in order to achieve even better performance in the next development iteration. However, when tuning the network



(a) Ordinary convolution with padding.

(b) Convolution with a dilation of two.

Figure 2.4: Illustration of convolution. The filter (gray) is moved across the input image (blue) with padding (white) to produce the convolved image (green). Images from Dumoulin and Visin [17].

to perform well on the validation set, there is a risk of adjusting too much to the validation set. Hence, a test set is used to validate the choice of best network model and its ability to generalize. This test set should not be used throughout the development process, only as a final check.

2.2.1 Training parameters

Training parameters, also called hyperparameters, control the training of a network, and can be tuned in order to achieve the best performance possible. In the following sections, a few network parameters are briefly described.

Optimizers

Each network uses an optimizer to determine how the weights should be updated for the next training iteration. A batch-size is specified for the network which determines how many training examples are used by the optimizer for the weight update. There are several well-known optimizers for image segmentation applications. Two examples are stochastic gradient descent (SGD) and Adam.

Stochastic gradient descent

The SGD optimizer updates the model parameters based on an estimate of the gradient computations of the risk (which measures performance in training), defined as

$$E_n(f) = \frac{1}{n}\sum_{i=1}^{n} Q(f(x_i), y_i), \tag{2.5}$$

where $Q(f(x), y)$ is the loss function, x is the input, y is the ground truth and f is the network function that minimizes the loss function [18]. The estimate of the process is computed for each training sample as

$$w_{i+1} = w_i - \eta_i \nabla_w Q(z_i, w_i), \tag{2.6}$$

where η is the learning rate, $z_i = (x_i, y_i)$ is a random sample and w is the weights [18, 19]. SGD takes more steps to reach convergence than most adaptive optimizers, but can have a smaller

final error and better generalization performance [20, 21].

Adam

The Adam (adaptive moment estimation) optimizer is a gradient-based algorithm, first introduced by Kingma and Ba [22]. It updates the weights depending on the first-order-gradient of the loss function as well as the estimates of first and second moments of the gradient. Hyperparameters for the algorithm is the step-size, also called learning rate, and exponential decay rates for the moment estimates. These parameters determine how much influence previous iterations have on the weight updates. Each weight is updated individually.

The Adam optimizer is defined by

$$\begin{aligned} t &\leftarrow t+1 \\ g_t &\leftarrow \nabla_w Q_t(w_{t-1}) \\ m_t &\leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t \\ v_t &\leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t^2 \\ \hat{m}_t &\leftarrow \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &\leftarrow \frac{v_t}{1-\beta_2^t} \\ w_t &\leftarrow w_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{(\hat{v}_t)} + \epsilon}, \end{aligned} \tag{2.7}$$

which is iterated until w_t has converged. In (2.7), t is the evaluated time step, g_t is the gradient with respect to the weights, the step size is α, the parameters β_1, β_2 are the exponential decay rates, and m, v are the moment estimates [22]. Adam is often used due to its fast convergence and as it often requires less tuning of the learning rate [21].

Learning rate

The learning rate regulates how large the weights updates are during an iteration. It is normally set to a value between 0.0 and 1.0. A large learning rate indicates a potentially large change in parameter values in every iteration, whereas a small learning rate causes a smaller change. Using a large learning rate can move the network faster towards convergence, but does not guarantee that it will converge [23]. Kingma and Ba [22] recommends a learning rate of 0.001 as starting point when using Adam.

Loss function

The loss function is a differentiable function which indicates how well the network performs. It needs to be differentiable so that the optimizer can calculate the gradient that is used for the weight update. One example is the Dice loss, defined by

$$L = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}, \tag{2.8}$$

for a binary segmentation. In (2.8), N is the number of pixels and p_i is the probability that pixel i belongs to the segment. The variable g_i is the ground truth (either 0 or 1) for pixel i [6].

Activation function

A common approach is to use rectified linear units (ReLUs) as activation functions. The activation function is described by

$$f(x) = max\{0, x\}, \tag{2.9}$$

which outputs a linear function for $x \geqslant 0$ and 0 when the input is negative. The function is illustrated in Figure 2.5. ReLUs are believed to speed up convergence compared to activation functions with a defined upper bound such as Sigmoid and Tanh [24].

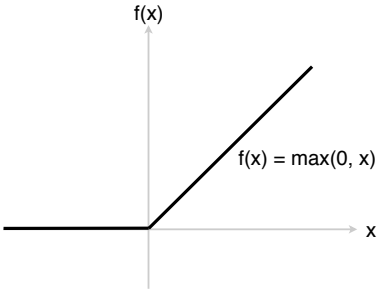


Figure 2.5: The ReLU function.

Number of iterations

The number of iterations is a hyperparameter which decides how many times the network should update the weights. Too few iterations could result in a network that has not learned suitable weights and thus performs poorly on both the training and validation sets. This is known as underfitting. Too many iterations on the other hand, can lead to overfitting, which means that the network learns the weights too well. An overfitted network performs well on the training set but does not generalize well, thus performs badly on a validation or test set.

Regularization

Regularization in general refers to any change made with the purpose to reduce overfitting. One method is L^2 regularization, which introduces weight decay during training. The modified loss function, J, to minimize is the sum of the original loss function, Q, and the L^2 norm, defined as

$$J(w) = Q + \lambda w^T w. \tag{2.10}$$

In (2.10), λ is the factor that decreases the weight influence [15, Chapter 5, 7].

Dropout is another regularization approach, which drops hidden nodes in the network randomly with a certain probability. In other words, a certain fraction of the nodes are not contributing to the summed output of the ANN during an iteration if dropout is used.

Batch Normalization

When training DNNs, the problem of vanishing and exploding gradients are well known. They occur either as large parameter changes leading to a small (vanishing) gradient or as small changes being amplified and resulting in a very large (exploding) gradient. Ioffe and Szegedy [25] introduced batch normalization to reduce these problems by normalizing the input to a

layer. Batch normalization also allows a higher learning rate, so that fewer training iterations are needed for convergence.

2.2.2 Data augmentation

Access to medical images with expert annotations are typically scarce, and in the process of developing deep learning algorithms this issue needs to be addressed. A common way of doing so is by applying data augmentation techniques.

Data augmentation modifies the available data in different ways to create "new" data. In the context of increasing the number of available images, new images can be created by rotating the original image, changing the gray levels etc. The aim is to expose the network for a greater variety of images and prevent it from overfitting to the training set. Augmentation can be applied online or offline. Offline augmentation means that augmentation is performed before training starts, thus increasing the size of the stored training set. Online augmentation on the other hand, will perturb the images "on the fly", meaning that an image is augmented between iterations during training. Online augmentation saves memory and opens up for more augmentation options, but might give less control and increase the training time.

2.2.3 Evaluation metric

One or several evaluation metrics are used to quantitatively evaluate a network's performance. It can be used during training to monitor the performance both on training and validation examples, but more importantly, for the final model's inference performance. In medical image segmentation tasks, there are often several labels, i.e., structures, to be segmented. For these tasks, the Dice score (DSC) is a commonly used metric that is sometimes referred to as overlap index [26]. There are other metrics that are used alone or in combination with DSC, such as the Hausdorff distance [27], which is distance based instead of overlap based.

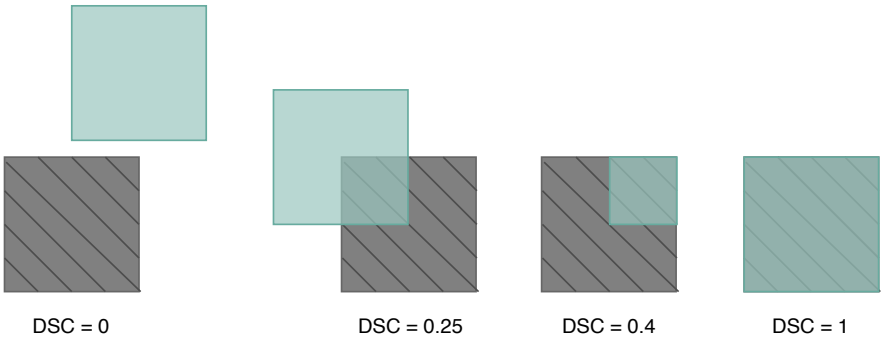


Figure 2.6: Illustration of DSC.

Dice Score

The Dice score measures the overlap between two segments, X and Y, and is defined as

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}. \tag{2.11}$$

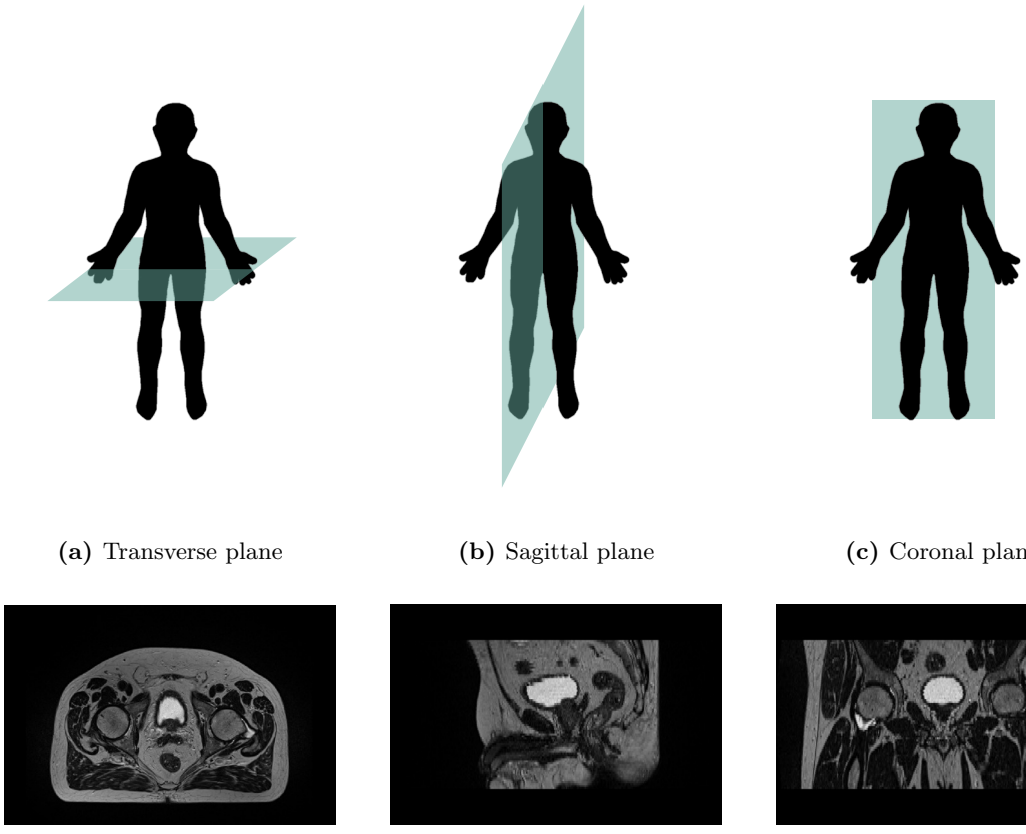
where $|\cdot|$ is the number of pixels of the specified segment. Using (2.11), it is straightforward to compare a network prediction with the ground truth. DSC is a dimensionless measure bound

between 0 and 1. A perfect prediction will generate a Dice score of 1, and no overlap at all generates a Dice score of 0. See Figure 2.6 for an illustration.

The Dice score as defined in (2.11) works for one label at a time, thus is only directly applicable to binary segmentation evaluation. There are different suggestions on how to deal with multi-label segmentation, for example averaging the DSC over all labels, or using a generalized Dice score which weights the different labels [28].

2.3 Magnetic Resonance Imaging

MRI is one type of medical imaging modality which differs significantly from other modalities due to the variability of the tissue characteristics that can be imaged [29]. An MR scan produces a stack of two-dimensional images, where each image is called a slice, showing the inner anatomy of the body. This image stack forms a volume which can be transected by different planes. The standard planes are the transverse plane, dividing the body into an upper and a lower part; the sagittal plane, creating a right and a left part; and the coronal plane, dividing the body into front and back. Figure 2.7 illustrates these concepts.



(a) Transverse plane **(b)** Sagittal plane **(c)** Coronal plane

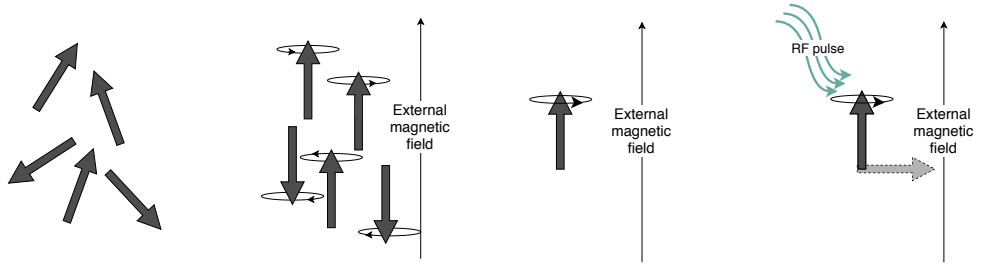
(d) Transverse MR slice **(e)** Sagittal MR slice **(f)** Coronal MR slice

Figure 2.7: The anatomical planes: transverse, sagittal, and coronal with MR image[1] examples.

[1]Original MR image courtesy of Gentle Radiotherapy

2.3.1 Technology

MRI is based on the magnetic properties of protons (hydrogen atoms), which are present in the entire body. The protons can be thought of as compass needles, each with an arrowed axis from its south to north pole. In the absence of an external magnetic field the poles of the protons are randomly located, pointing in all different directions, see Figure 2.8a. However, placing the body part of interest in an external magnetic field will make the protons align with or against the external field direction (Figure 2.8b). The protons have a slightly higher probability to align with the field than against it, resulting in a net magnetization vector (Figure 2.8c). Tissues with higher proton density (PD) will consequently build up a higher net magnetization vector than tissues with a lower number of protons. Moreover, when placed in an external magnetic field, the protons will begin to precess around the magnetic field axis with a specific frequency known as the Larmor frequency.



(a) Randomly oriented protons in the absence of an external magnetic field.

(b) Protons aligned with or against the external magnetic field. The protons precess around the magnetic field axis.

(c) The protons form a net magnetization vector.

(d) Applying an RF pulse tips the net magnetization vector.

Figure 2.8: The basis for MRI technology.

An external radio frequency (RF) pulse with the same frequency as the protons' Larmor frequency can be applied to affect the protons in the magnetic field. The magnetization vector is tipped to the plane perpendicular to the external magnetic field (Figure 2.8d). This will produce a detectable signal with the Larmor frequency.

Applying field gradients in the x-, y- and z-direction enables spatial encoding, i.e., being able to extract voxel by voxel data. The gradients are switched on at different times during the scanning procedure. One of the gradients is used to select which slice to image. For each selected slice, a so called k-space is created, containing information about frequency, phase, and signal intensity from that slice. The image data can be extracted using the Fourier transform.

As previously mentioned, MRI can image different tissue characteristics. The different characteristics are PD, T1 (longitudinal relaxation time), and T2 (transverse relaxation time). T1 and T2 describes how fast the protons return to their normal state. Which tissue characteristic, or combination of characteristics, will be depicted is controlled by the pulse sequence. The pulse sequence determines RF pulse length and frequency, as well as time between pulses. One image type that can be generated is the T2-weighted image, where differences in T2 between tissues give rise to contrast in the image [29]. Structures with long T2 relaxation time, such as fluids, are depicted as bright, whereas rigid structures, such as bones, are dark in these images.

2.3.2 Image quality

The image quality of an MR image can be specified by the level of detail and visible noise in the image. Decreasing the voxel size increases the level of detail which can be imaged. However, decreasing the voxel size means fewer protons contributing to signal intensity and thus lowers signal-to-noise ratio (SNR). SNR is also affected by magnetic field strength and the pulse sequence. Higher magnetic field strength generally results in higher SNRs [29].

Where there is no MR signal, e.g., in the air, the noise can be estimated by a Rayleigh distribution. Where the SNR is high the noise distribution approximates the Gaussian distribution. Elsewhere, the noise can be modeled by a Rician distribution, as described by Gudbjartsson and Patz [30]. The different distributions are illustrated in Figure 2.9.

The probability density function (PDF) of the Gaussian distribution is

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(x-\mu)^2}{2\sigma^2}}, \tag{2.12}$$

where μ is the mean of the PDF and σ is the standard deviation [31].

The PDF of the Rayleigh distribution is

$$f(x) = \frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}, \tag{2.13}$$

for $x \geq 0$ with σ as the scale parameter [32].

The Rician PDF is

$$f(x) = \frac{x}{\sigma^2} e^{\frac{-(x^2+\nu^2)}{2\sigma^2}} I_0\left(\frac{x\nu}{\sigma^2}\right), \tag{2.14}$$

where I_0 is the zero-order Bessel function of the first kind [33] and σ, ν are shape parameters.

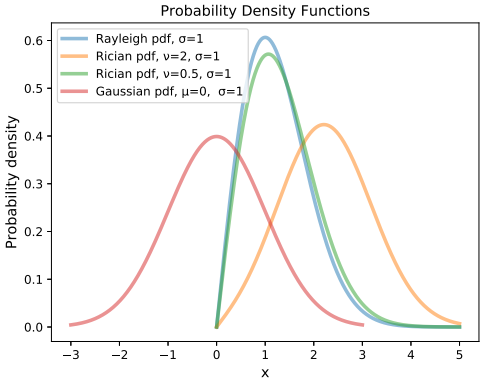


Figure 2.9: Examples of probability density functions of the Gaussian, Rayleigh, and Rician distributions.

Another type of disturbance in MRI is the bias field. The bias field is a slow intensity variation across the image stack which can be modeled as a multiplicative effect [34]. The distortion is not always observable to the human eye.

MRI has a rather long scanning time. A full MR scan, which results in different types of image volumes, takes 20 - 45 minutes [35]. The long scanning times give rise to motion artifacts. For example, breathing artifacts will show as "rings" due to the chest moving up and down during acquisition.

2.3.3 Multi-site problems

When using MR scans from different hospitals and instances, the data set is commonly referred to as multi-site data. The problem with using multi-site MR data originates from the inter- and intra-variability in MR machines. Intra-variability over time can be due to exchange of hardware and upgrades of software [36]. Inter-variability originates from different vendors as well as system performances [37]. Different approaches to deal with this variability have been developed and described in literature for the purpose of conducting large-scale multi-site studies [36, 38].

Multi-site problems are also relevant in the context of DL-based organ segmentations. A CNN which has been trained on single-site MR data does not necessarily, and is not likely to, perform well on data from another site, especially not if the image characteristics differ significantly. These differences might not affect a human annotator but is interpreted differently by a network.

2.4 Radiotherapy planning

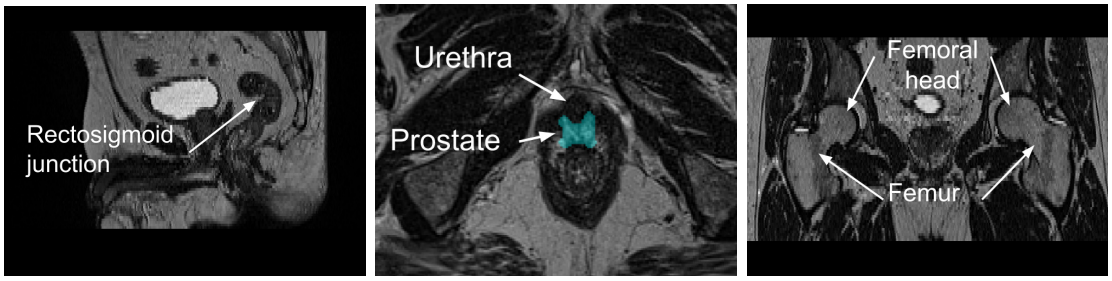
The intention of an automatic organ segmentation algorithm is for it to be clinically useful, either by improving the patient outcome, easing the workload of clinicians, or both. When a patient is planned for radiotherapy, a CT, MR image, or both, is taken in order to correctly plan the treatment. The image stacks are reviewed by doctors or medical dosimetrists who manually delineate the OARs and target. Then, a dose matrix is set up, determining how much radiation that should be given from each angle (360° around the body). Each angular dose depends on the segmentation and potential restrictions, such as a disease or prosthesis, which limit the allowed radiation from certain angles. The aim of the radiotherapy is to irradiate the tumor with as high dose as possible, while sparing the surrounding tissue. In a clinical setting the terms clinical target volume (CTV) is the prostate (with or without seminal vesicles depending on tumor location) [39]. In order to allow for uncertainties in the planning procedure and dose delivery, the CTV is extended to the planning target volume (PTV), which makes sure that the target receives a sufficiently high dose [40]. If not, the tumor remains and the radiotherapy only caused damage to the surrounding tissue.

For each OAR there are dose restrictions depending on volume, absolute dose, or both, which are taken into account before the treatment plan is set. The exact dose each OAR receives is dependent on where the organ is located and the ability for the different tissues to stop the radiation, i.e., stopping power. The gray level intensities in the CT image can be directly translated to the stopping power of each voxel, which is not possible using the MR image. This is due to the difference in signal generation of the images. On the other hand, MR images are superior in soft-tissue contrast and are often used as a complement to the CT in the delineation process.

Using multiple imaging modalities brings issues such as additional imaging time and planning, as well as organ movement between image acquisitions. Therefore, it is of interest to use a single modality, where MRI-only treatment is a promising next step with both reduced patient costs and radiation doses for OARs [41].

2.4.1 Manual segmentation

The manual segmentation of prostate and OARs are known to be subject to inter-expert variability and has been reported in literature [42]. Salembier et al. [43] sought to create guidelines for contouring prostate and rectum to decrease the variability. With 11 experts, of which ten were radiation oncologists and one was a radiologist, the ESTRO guidelines were formed. The



(a) Rectosigmoid junction in the sagittal plane.

(b) Prostate in the transverse plane segmented with butterfly shape in apex to exclude the urethra.

(c) Femur and femoral head in the coronal plane.

Figure 2.10: Examples of anatomical structures in MR images[1] *used in manual segmentation.*

guidelines recommend that the rectum is segmented from the rectosigmoid junction to approximately two centimeters below the prostate apex (lower part of the prostate). The prostate apex should have a butterfly shape that excludes the urethra. The prostate is connected to the rectum at mid-level and to the bladder at the base (upper part of the prostate). Seminal vesicles are included if at risk of tumor invasion. Figure 2.10a shows the rectosigmoid junction, and Figure 2.10b shows the butterfly shape in prostate apex.

Contouring of the bladder is not defined in the ESTRO guidelines, but to our knowledge usually includes both the bladder and the bladder wall, as in the study by Thörnqvist et al. [44].

The segmentation of femur vary depending on how much of the femur bones is included in this OAR delineation, as some delineate only the femoral head while some include more of the femur [45]. The femur in the coronal plane is visualized in Figure 2.10c.

Table 2.1: Variability between experts when conducting a cross-evaluation on achieved dice scores.

Organ	**Mean**	**SD**	**Min**
Bladder	0.9298	0.0308	0.8242
Rectum	0.8370	0.0984	0.4940
Left femur	0.9166	0.0274	0.8174
Right femur	0.9166	0.0291	0.7841

2.4.2 Inter-expert variability in manual segmentation

Organ delineation is not an easy task, not even for experts, which gives rise to a variability. To quantify this, Spectronic Medical have evaluated five experts' segmentations from Gentle Radiotherapy [8] for bladder, rectum, and femur delineations for 11 patient cases. In the study, one annotator's work was considered ground truth and compared to the others' segmentations. The procedure was repeated for each annotator, as illustrated in Figure 2.11. The study confirms that OAR segmentation differs depending on the annotator. The average dice score for each organ is presented in Table 2.1. From these numbers, it seems that the experts are fairly

[1]Original MR images courtesy of Gentle Radiotherapy

consistent in bladder and femur segmentation with average dice scores of $0.917 - 0.930$, but inconsistent for rectum with an average dice of 0.837 and a standard deviation of 0.0984.

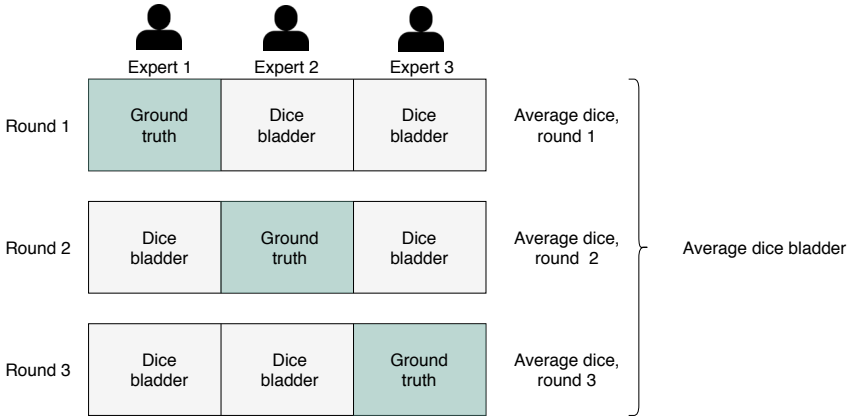


Figure 2.11: Design of the inter-expert variability study. This only shows an evaluation for bladder and three experts, but the same principle was used for all organs in the variability study with five experts.

Chapter 3

Data and other premises

This chapter describes the available data set and the differences between images from different sites. The software and hardware which was used is also described, as it provided both possibilities and limitations to the work.

3.1 Data

The data set used in this work contained T2-weighted MR images from 151 patients diagnosed with prostate cancer. The image volumes have a large field of view (FOV) which in most cases covers the entire width of the patient. The volumes were collected from four hospitals around Sweden, each hospital with different MR cameras, see Table 3.1.

The inclusion criteria for the study were not strict since a varied data set was preferred. Therefore, one MR scan was included which does not have a large enough FOV to capture the entire body in the transverse plane. The majority of the MR images are of good quality, but a few have low SNRs or contain breathing artifacts. For some patients, the organs are hard to visually separate. For example, in one image volume, the prostate is deviant in size, which makes the prostate and rectum seem fused with no clear boundary between them.

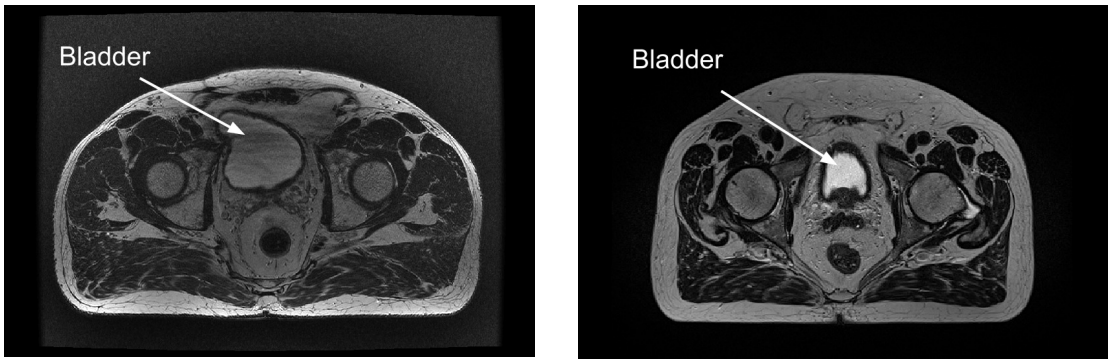
Table 3.1: Site-specific MR parameters.

	Parameters					
Site	**No. of patients**	**MR scanner**	**Field Strength (T)**	**No. of slices**	**Slice thickness (mm)**	**Spatial resolution*, transverse plane (mm)**
A	56	GE Discovery	3.0	71-74	2.5	0.438-0.457
B	47	GE Signa PET/MR	3.0	68-86	2.5	0.398-0.457
C	42	Siemens Aera	1.5	78	2.5	0.875
D	6	Siemens Skyra	3.0	78	2.5	0.869

** Reconstructed spatial resolution.*

No expert segmentations were available as ground truth for this data set. Hence, the organs were delineated by the authors of this report. This was based on a previously trained segmentation network at Spectronic Medical and aided by instructions from a Gold Atlas for

prostate segmentation, conversations with an expert at Skåne University Hospital (SUS), and a synthetic CT (sCT). The sCT was used to correctly segment the bones and was created by Spectronic Medical's software MriPlanner [46]. For more details on the segmentation process, see Section 4.1.



(a) Example of MR image slice from site A

(b) Example of MR image slice from site C

Figure 3.1: Comparison of MR image slices from site A and C.[1] *Notice the difference in noise in the background and the gray level intensities in the bladder between the slices. Furthermore, the intensity variation in the adipose tissue, created by a bias field, is more distinct in the left slice than in the right.*

Each of the 151 MR image volumes had been preprocessed before handed to us through individual mean and variance normalization of the pixel intensities and removal of the outermost slices. These slices had a smaller FOV than the rest of the slices in the stack. After preprocessing, each MR image stack consisted of 68 - 86 transverse slices. For further characteristics, see Table 3.1. It should be noted that all sites except site B had fairly consistent parameter values for all their MR images.

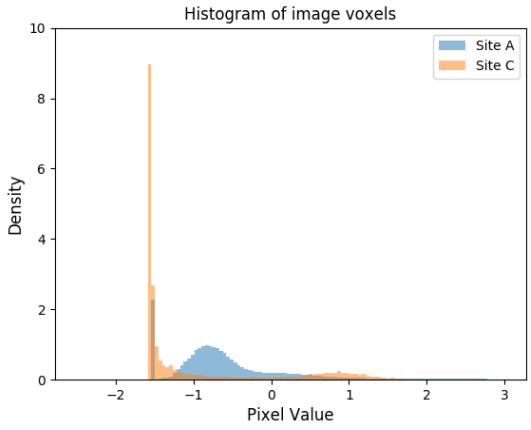
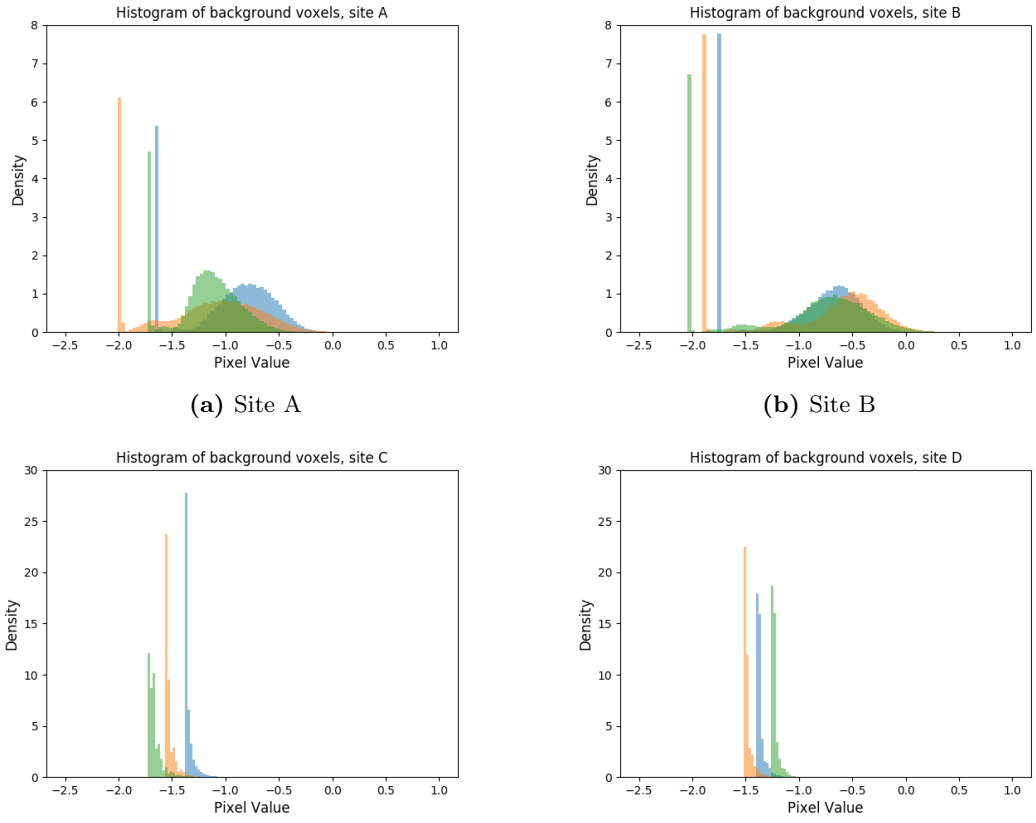


Figure 3.2: Histograms of the two MR images in Figure 3.1. For site C, the noise-free background gives rise to the peak at -1.5 and the bright bladder a small rise at 1.

[1] Original MR images courtesy of Gentle Radiotherapy

The MR images have different characteristics depending on which site they were collected at since the camera properties and scanning protocols differ. Some characteristics are possible to distinguish visually, here illustrated in Figure 3.1, where noise level and the bladder intensities are two key differences. Other characteristics are most likely present but not as easy to visually distinguish. The corresponding histograms for the two image volumes exemplified above can be seen in Figure 3.2, where the difference in intensity distribution is illustrated.

To further emphasize the difference in noise levels between sites, histograms of the background pixels for three random image volumes from each of the four sites are presented in Figure 3.3. Site A and B which have GE scanners, show higher noise levels than site C and D, which have Siemens scanners. As can be seen, the histograms are located around different pixel values. Due to the preproccesing through mean and variance normalization, some of the pixel values are negative, which is not possible in a non-processed MR image. This should not confuse the reader, but what should be noted is the overall shape of the histograms.



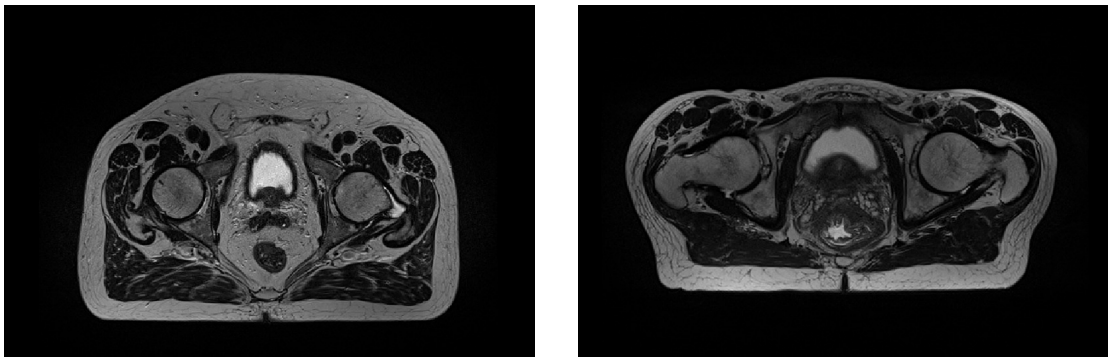
(a) Site A **(b)** Site B

(c) Site C **(d)** Site D

Figure 3.3: Examples of histograms of background voxels for three random image volumes from each site, depicted in green, orange, and blue. Note that each histograms represents one image volume, i.e., histograms of 12 different patients are shown.

The differences are not only present between vendors, but also between sites, as can be seen in Figure 3.4 where the bladder from Site C is much brighter compared to the bladder from

Site D.



(a) Example of MR image slice from site C

(b) Example of MR image slice from site D

Figure 3.4: Illustration of difference between MR images[2] from same vendor but different hospitals. Notice the difference in bladder intensity.

The MR images and the manually performed segmentations constitute the data set used for training the networks and evaluating their performance. More information on how the segmentations were created is described in Section 4.1.

3.2 Software

This work has been performed in the Python programming language. Throughout this thesis, the NiftyNet 0.4.0 framework was used. NiftyNet is an open-source platform used for image analysis within the deep learning field and is built on Tensorflow [47]. To add or modify NiftyNet, SimpleITK [48, 49] was utilized for many image analysis tasks. The open-source segmentation tool ITKSnap [50] was used for organ delineation.

3.2.1 NiftyNet

NiftyNet is a framework, introduced by Gibson et al. [51], for building CNN applications which handles all sorts of tasks from loading and sampling an image volume, to data augmentation and fast optimization. The framework has applications ready for use and consists of a collection of Python script files. The user can modify the existing files or add new ones in order to customize the applications.

The modules of the network are initialized by the Application driver which divides tasks according to the available hardware and keeps track of the ongoing processes. The main processes are in turn divided into a sampler, a training iterator, and an inference iterator. As a training is initiated, the order of the training image volumes is randomized. Then, the sampler reads input image volumes, adds padding, and preprocesses them before potential augmentation is applied. The ground truth is also padded and augmented geometrically. Thereafter, a number of smaller image volumes are extracted from the original volume and put in a queue, see Figure 3.5. The number of samples drawn from each image stack reflects how many times the weights are updated based on the same stack.

[1] Original MR image courtesy of Gentle Radiotherapy

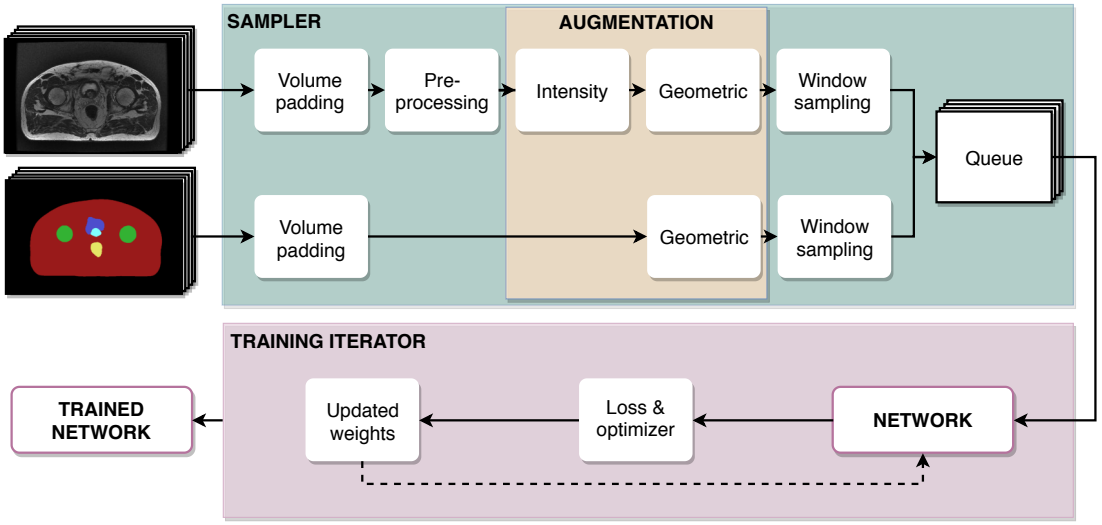


Figure 3.5: An overview of the sampler and training iterator in NiftyNet.[1]

Each available graphical processing unit (GPU) takes one batch of images at a time from the queue and passes it through the network. The training iterator works as previously explained in Section 2.2: the proposed segmentation is compared to the ground truth and a loss is calculated. Based on the loss value, the weights are updated in the network. Once all iterations are done, a trained network is obtained which can be used for inference.

During inference, an image volume is read, padded, and preprocessed before the entire volume is sampled. The window samples are put in the queue and separately passed through the network. The resulting patches are collected by the aggregator which pieces the window segmentations together before outputting the final segmentation. For an illustration of the inference flow, see Figure 3.6.

The network settings are specified in a configuration file, where parameters such as learning rate, optimizer type, and what kind of augmentations to use are set.

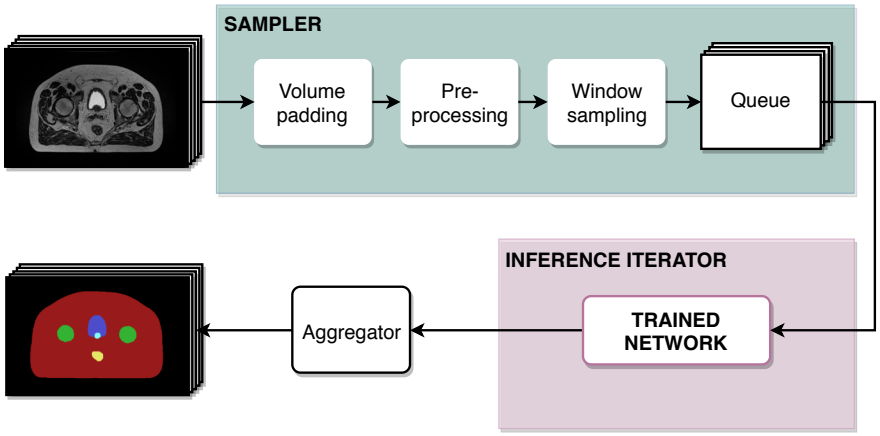


Figure 3.6: An overview of the inference iterator in NiftyNet.[1]

[1]Original MR image courtesy of Gentle Radiotherapy

Network architecture

One of the implemented CNN architectures in NiftyNet is *HighRes3DNet_large*. The net was originally designed by Li et al. [52] to be used in segmentation tasks involving fine structures.

It is used directly or with slight modifications in numerous image analysis applications [53, 54]. The version that is implemented in NiftyNet is composed of several combinations of blocks with three components: convolutions, batch normalization, and an activation function. The convolutional blocks are $3 \times 3 \times 3$ convolutions with 16 - 64 kernels and dilation by 1, 2, or 4, except for the two last blocks which are $1 \times 1 \times 1$ convolution with 64 kernels and the same number of kernels as of classes to segment, respectively [51]. The network has rectified linear units (ReLU) as activation functions [52].

All $3 \times 3 \times 3$ convolutional blocks, except the first and last, are combined two and two, to form residual connections. These connections allow fusion of information at different scales and potential to reduce training time [52]. The structure is depicted in Figure 3.7.

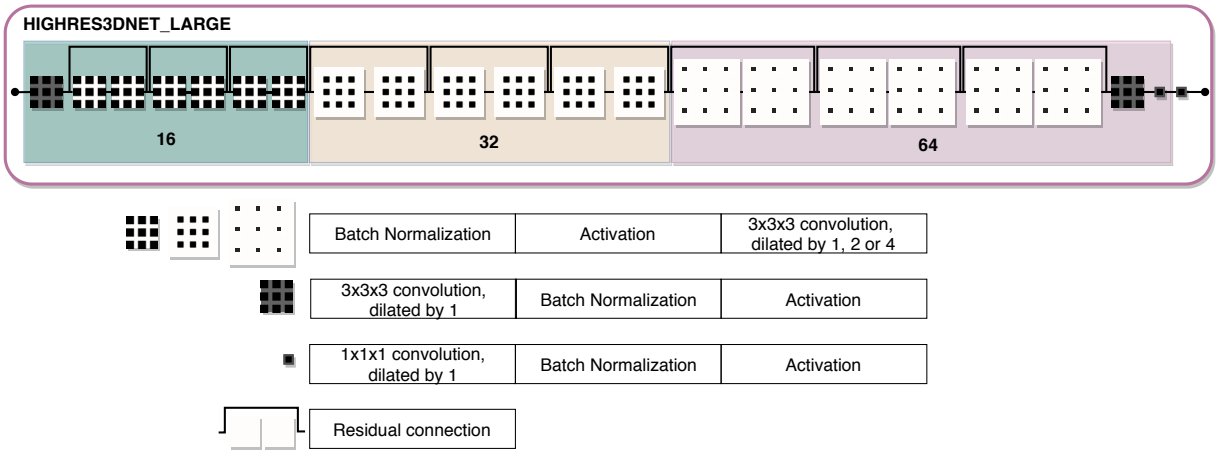


Figure 3.7: The components of the HighRes3DNet_large network. The numbers 16, 32, and 64 illustrates the number of kernels for each layer. The last layer has the same amount of kernels as number of classes in the output. For the $3 \times 3 \times 3$ blocks, the spacing in the grid illustrates the dilation factor (1, 2, or 4) where a larger spacing indicates a larger dilation.

3.3 Hardware

During this thesis, the network trainings were conducted on computers with a NVIDIA GeForce GTX 1080, NVIDIA Titan RTX, or NVIDIA Tesla graphics card. For more information, see Table 3.2. The computers with NVIDIA Tesla's were accessed through a prostate segmentation project funded by Analytic Imaging Diagnostics Arena (AIDA) [2].

Table 3.2: NVIDIA graphics card properties.

Graphics card	No. of GPUs	Memory (Gb)
NVIDIA GeForce GTX 1080	1	11
NVIDIA Titan RTX	2	24 + 24
NVIDIA Tesla P40	1	22.9
NVIDIA Tesla P40	1	24.5

Chapter 4

Methods

The following section presents the used methods in this thesis. First, the segmentation process is explained in detail, followed by how the networks have been evaluated. Thereafter, the experiment set up and hyperparameter tuning is described. Lastly, the augmentation techniques are presented, both the ones already available in the training framework and new ones developed during this thesis.

4.1 Manual segmentation

As mentioned in Section 3.1, the ground truth segmentations for the 151 MR image stacks were delineated by the authors of this report, since no expert annotations were available. The segmentations were conducted in ITKSnap on the preprocessed T2-weighted images.

The guidelines presented in Section 2.4.1, together with a Gold Atlas for OAR segmentation from *Gentle Radiotherapy* [8] and discussions with Adalsteinn Gunnlaugsson, doctor at the department of Oncology and Pathology at SUS Lund, served as basis for the segmentation. To gain insight in prostate segmentation, we were instructed by Gunnlaugsson during a visit to the hospital. He presented the prostate anatomy and the guidelines the doctors at SUS follow for target delineation, which aligns with the ESTRO guidelines.

When delineating the prostate, the risk for invasion and tumor characteristics are deciding factors in whether seminal vesicles should be included in the segmentation or not. This information is provided in patient journals which have not been available for this segmentation task. Therefore, the seminal vesicles were not included in any prostate segmentations.

When delineating the femur, only the femoral heads were included, and a circular shape was aimed for in all three dimensions. The sCT aided the segmentation of the femoral heads as it is easier to distinguish the cortical bone in the sCT than in the MR image.

The anal canal and rectum, here segmented as one organ, was segmented to the recto-sigmoid junction as stated in Section 2.4.1. A volumetric restriction was also applied, striving for a rectum segment to stretch about ten centimeters in the sagittal plane.

The whole bladder, together with the bladder wall, was segmented. In some cases, the bladder can be seen with a downward facing "peak" into the prostate. This is due to a medical procedure, and was segmented as prostate.

The body contour segmentation was done from an initial thresholding of the sCT, as it has a larger contrast between body and air, and SNR is much higher than in the MR image. The contour was later reviewed and corrected where needed.

Both authors took part in the segmentation of MR images. Each image stack was delineated by one person, then reviewed by the other. Any uncertainties were discussed to ensure consensus within this work.

4.2 Evaluating network performance

Each trained network was evaluated quantitatively and qualitatively. First, a quantitative analysis was made using the DSC. For each patient, DSCs were calculated for each organ and an average was taken of the so called foreground labels, referring to the body, femur heads, bladder, rectum, and prostate. The average score for all patients in the validation set was then averaged and used as single evaluation metric of the network performance.

To complement the quantitative evaluation, a qualitative assessment was made through visual examination of the network segmentations. Random validation samples were used for evaluation of general performance and coverage of each organ. This was also used to ensure that DSC continued to represent a good mapping of segmentation accuracy.

4.3 Experiment design

In order to answer the questions posed in Section 1.2, the experiments presented in Section 4.3.1 - 4.3.3 were designed. The essential difference between the experiments is the site distribution of the training, validation and test set. The sets were randomly drawn with regards to some site restrictions explained in Sections 4.3.1 - 4.3.3. This partitioning was kept throughout the work. Table 4.1 provides an overview of the number of training, validation, and testing examples in each experiment. The training set size was chosen to be as large as possible at the same time as ensuring at least two subjects from each site in the validation or test set. Case 1 - 3 were evolved in parallel, trying to optimize the network for each unique case which is described in detail in Section 4.4.1. Throughout the thesis, a few additional case studies were conducted, which are presented in Appendix H.

Table 4.1: Overview of the number of MR image volumes from each site in the data sets used in the experiments.

	Training					**Validation**					**Test**				
Case	A	B	C	D		A	B	C	D		A	B	C	D	
1.1 and 2.1	40	0	0	0	**40**	13	37	35	5	**90**	3	5	12	1	**21**
1.2 and 2.2	0	0	40	0	**40**	43	41	2	4	**90**	13	6	0	2	**21**
3	41	30	26	3	**100**	11	11	12	2	**36**	4	6	4	1	**15**

4.3.1 Case 1 - Single site training

In order to establish a baseline, the training for case 1 took place without any augmentations. The training set consisted of 40 MR image volumes from a single site, the validation set of 90 volumes, and the test set of 21 volumes. There were two versions of this case, where case 1.1 only had image volumes from site A (GE camera) in the training set and case 1.2 only had image volumes from site C (Siemens camera).

4.3.2 Case 2 - Single site training with augmentation

Case 2 was created to investigate the effect of augmentation. In this case, training image stacks were modified during the training process through various augmentation techniques. The different augmentation techniques are further described in Section 4.5. Similar to case 1, this case has two versions, and the same data split for case 1.1 and 2.1 was used as well as for case 1.2 and 2.2.

4.3.3 Case 3 - Multi-site training

Case 3 was designed to give an indication of how well a network performs if the training set has image stacks from the same distribution of sites as the validation and test sets. Therefore, a large training set of 100 image volumes, a validation set of 36 volumes, and a test set of 15 volumes with all four sites represented in all sets were used for case 3.

4.3.4 Repeatability

There are many elements of randomness when training a network. For example, network weights are randomly initialized prior to training. In addition, random samples from the input image stakcs are generated, random augmentation layers and parameters are used during training. It came to our understanding that this resulted in a variability in performance when retraining a network with the same setting multiple times. To account for the performance variability and attempt to reach unambiguous conclusions, the final network configurations were used to train a network five times. This resulted in five different models and results for each case. The model with the highest mean DSC on the validation set was selected as the final model.

4.4 Network configuration

Since this thesis aims to investigate augmentation and its effects, the largest work with the networks have been configuring augmentation parameters and modifying or adding augmentation layers, rather than optimizing network architecture and other hyperparameters. Some tuning of the hyperparameters took place, which is described in Section 4.4.1. In this thesis, $HighRes3DNet\_large$ was selected as network structure and Adam as optimizer.

In order to use $HighRes3DNet\_large$ with the desired window sampling size on the available hardware, the image volumes were downsampled to the voxel size $2 \times 2 \times 2.5$ mm. This, in turn, determines the resolution for the output.

4.4.1 Tuning hyperparameters

Initially, all experiments had the same hyperparameter configuration. The hyperparameters were tuned by either trying to reduce the generalization error or increase the training performance, depending on the quantitative evaluation.

Each iteration of a case (not to confuse with a training iteration in the network) was evaluated on the validation set to determine the direction forward. Depending on the result of inference on the training set and the validation set different actions were taken. If the result on the training set was not satisfying, parameters such as learning rate and number of iterations were adjusted. If the training result was satisfying, but with a noticeably lower validation set performance, actions on reducing this variance were taken instead, such as adjusting dropout and other regularization parameters.

Table 4.2: A summary of the hyperparameters that vary between cases in the experiment set up.

	Case				
Parameters	**1.1**	**1.2**	**2.1**	**2.2**	**3**
Learning rate	0.0100	0.0100	0.0010	0.0010	0.0010
Learning rate decay factor	1/8	-	1/3	1/3	1/3
Normalization	False	True	False	False	False
Dropout	0.0	0.1	0.0	0.0	0.0

Both the quantitative and qualitative evaluation analysis was used to identify the more problematic network tasks. For example, bad segmentation performance on a specific organ, handling of breathing artifacts, and wrong placement of organs were noted. These observations were used to improve the next network's performance. The network with the highest DSC was considered the best and used as basis for the next network iteration. The final configurations were then used for the five repeated runs, as previously explained.

4.4.2 Network parameters

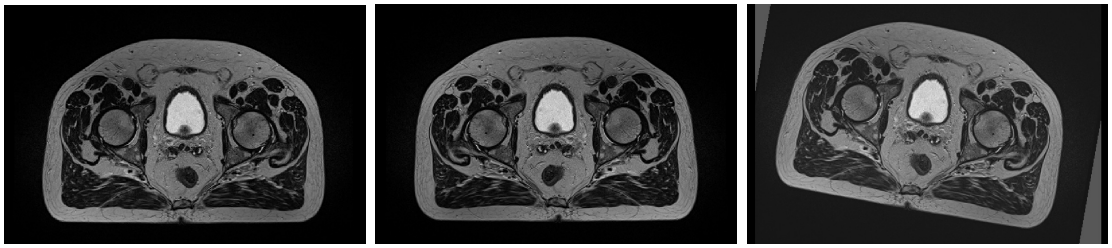
The networks for case 1 - 3 were trained with the same spatial window size, pixel dimensions and number of samples per read image volume, using a batch-size of two. Dice was used as loss type, the window sampling was uniform over the image volume and each volume was mean-variance normalized. The used activation function was ReLU and the regularization type was L^2 with 0.00001 as regularization term. All networks were trained for 30 000 iterations. The parameter values that vary between cases can be seen in Table 4.2. As indicated by the *Normalization* parameter, the input image stacks for case 1.2 were preprocessed by a histogram standardization method, in addition to the mean-variance normalization.

4.5 Augmentation

Initially, offline augmentation was used, but quickly abandoned for online augmentation due to the advantage of exposing the network to a greater variability of image stack distributions. For example, if augmenting in four different ways with an independent probability of 50% for the augmentation to occur, the probability that an image volume would not be augmented is 6.25%. During a training which runs 30 000 iterations, where each image volume is sampled and used for five iterations, the network is exposed to 6 000 volumes. Approximately 400 volumes will not be augmented, and 5 600 will be. To introduce this variation offline would require a substantial amount of memory.

Some online augmentation techniques were already available in the NiftyNet framework, while others had to be implemented. All different types of augmentations were implemented as layers in NiftyNet. The augmentation layers were applied to the full, padded image stack before sampling and placing the sample in the queue, see Figure 3.5. For details on the used parameter settings, see Section 4.5.6.

In this thesis, geometric and intensity-based augmentations have been tested and are described in the following sections. Some ideas for the augmentation techniques originated from the visual differences between the sites. However, it is worth noting that the aim of augmentation was to introduce a variability and not to resemble each site characteristic perfectly. The augmented image stacks were therefore not necessarily realistic, as seen in some of the exam-



(a) Original image slice. **(b)** Flipped image slice **(c)** Image slice rotated clockwise by 10 degrees

Figure 4.1: Example of geometric augmentations in MR images[1].

ples in this section. A reason for this approach was that it does not require information about the sites, and what the data from each site looks like. The expectation was that introducing additional sites, for which scarce data and information is provided, will not be a challenge for the trained network.

4.5.1 Geometric augmentations

Geometric augmentations moves the voxels of the image stack in space and are therefore applied both to the MR image and the ground truth. In this work, rotation, flipping, scaling, and elastic deformation have been used. These augmentation layers are implemented and ready to use within NiftyNet.

The flipping layers can flip an image volume around any of the three axes. The user selects which of these three axes that should be possible to flip. It is possible for more than one axis to be flipped in one augmentation. Figure 4.1b illustrates flipping the left-right axis.

The rotation layer takes an argument for rotation angle range limits. For each image stack, three random numbers are drawn from the range, one for each dimension. The numbers determine the degree of rotation. Figure 4.1c shows a rotation of 10.0 degrees.

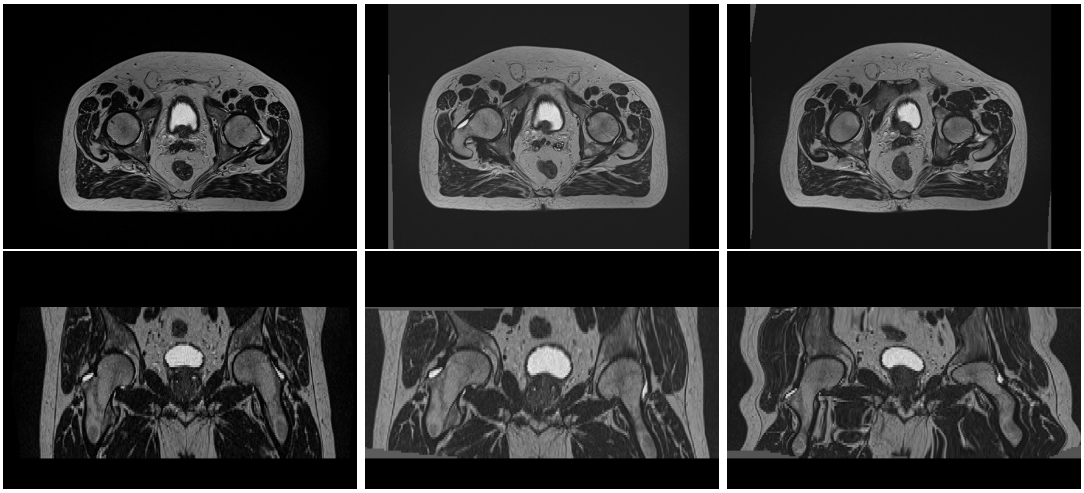
The image volume size can be modified by using the scaling layer. It draws a random scaling percentage from a user-specified interval which is applied in all dimensions. The scaling percentage is relative to the original image volume size.

The elastic deformation layer is more complex than the other geometric augmentation layers. The algorithm transforms the original image stack using the B-spline transformation as implemented in SimpleITK [55]. A B-spline is defined by its control points. The augmentation layer modifies the initialized control points at each augmentation in a random way defined by a user-specified standard deviation. Figure 4.2 illustrates the effect of elastic deformation using 3 or 8 control points, respectively, and a standard deviation of 15.

4.5.2 Histogram-based augmentations

A histogram-based augmentation layer was developed using some already implemented helper functions in NiftyNet. The layer modifies the histogram of the image stack voxel values to produce a new image stack. The algorithm is explained partly in this section, and with more details in Appendix B.

[1] Original MR image courtesy of Gentle Radiotherapy

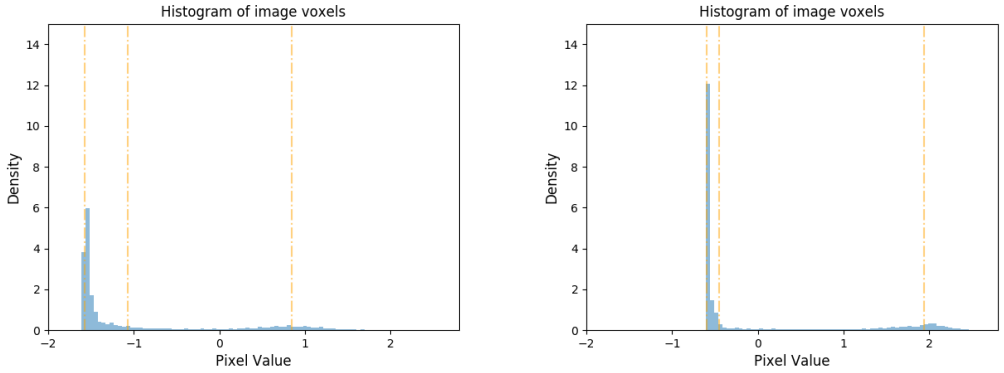


(a) Original image slice.

(b) Deformation result when using three control points.

(c) Deformation result when using eight control points.

Figure 4.2: Result of elastically deforming the same original MR image[1] with two different number of control points. The deformation in the transverse plane is found in the upper row, and the coronal plane in the lower row.



(a) Original image slice.

(b) Augmented image slice.

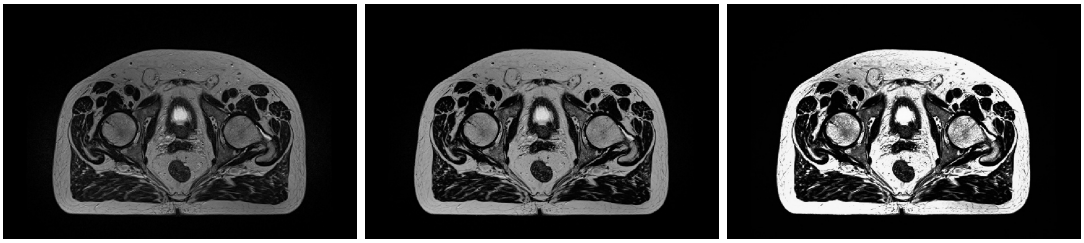
Figure 4.3: Histogram modification by randomly shifting percentiles. The yellow, dashed lines indicate the 10th, 70th, and 90th percentiles from left to right.

The percentiles $[0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95]$ of the image stack histogram are calculated, see Figure 4.3a. Thereafter, they are scaled to the range $[0, 100]$. Each percentile, p_i, is modified to $p_{i,shifted}$ by

$$p_{i,shifted} = p_i + \mathcal{N}(0, p_i\alpha), \tag{4.1}$$

where α is a user-specified histogram modification factor. Any values that fall outside the range will be replaced by a random number between 0 and 1, for values below 0, and between 99 and

[1]Original MR image courtesy of Gentle Radiotherapy



(a) Original image slice.

(b) Augmented image slice corresponding to the histogram in Figure 4.3.

(c) Another histogram augmented image slice.

Figure 4.4: Two examples of histogram augmented images originating from the same original MR image[1].

100, for values above 100. The new values are sorted and represent the percentiles of the pixel values in the augmented image stack.

The original image stack is mapped by a linear transformation procedure, defined in NiftyNet [56], to match the new percentiles. The procedure maps each percentile of the original image stack linearly to the modified percentile. Hence, there will be one linear function for each percentile. Each voxel in the original image volume belongs to a percentile, which determines which of the linear functions will be used to map the voxel value to a new value. The mapped image stack, x_{map}, is normalized as

$$x_{aug} = \frac{x_{map} - \bar{x}_{map}}{\sigma_{xmap}}, \tag{4.2}$$

to produce the final augmented image stack, x_{aug}. In (4.2), $\bar{x}_{map}$ is the mean pixel value, and σ_{xmap} is the standard deviation of pixel values.

Figure 4.3 shows how the histogram and the 10th, 70th, and 90th percentiles have been modified through this method. Figure 4.4 visualizes different results of applying the histogram modification layer twice on the same original MR image.

4.5.3 Bias field augmentation

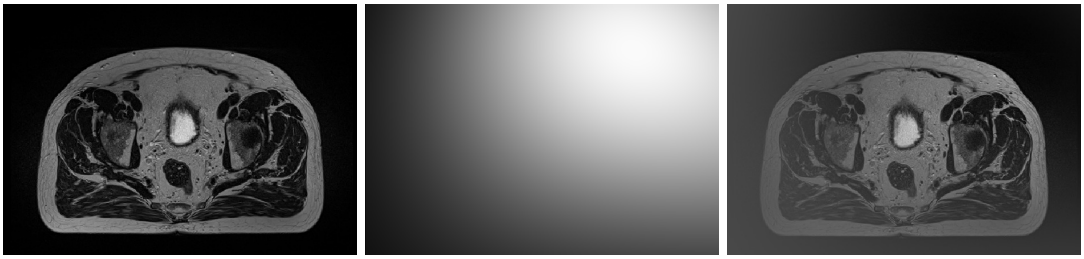
A bias field augmentation which simulates an intensity variation over the image volume was available in the NiftyNet framework. A grid of the same size as the image volume is created. Each voxel (x, y, z) is assigned a value ranging from [-1, 1]. A map, bf_{map}, is created as

$$bf_{map}(x, y, z) = \sum_{i=0}^{m} \sum_{j=0}^{m-i} \sum_{k=0}^{m-(i+j)} r_{i,j,k} x^i y^j z^k, \tag{4.3}$$

where $r_{i,j,k}$ are randomly drawn coefficients within a user-specified range. In (4.3), the integers $i, j, k \, \epsilon \, [0, m]$, where m is the specified order as determined by the user. The map is multiplied with the original image stack to produce an augmented image stack with intensity variations. An example of one slice from the bias field volume (4.5b), and the effect of multiplying it directly with an image stack, is shown in Figure 4.5c.

Since the used MR images have been preprocessed through mean and variance normalization, there are both negative and non-negative pixel values in the image volume. This leads to that the

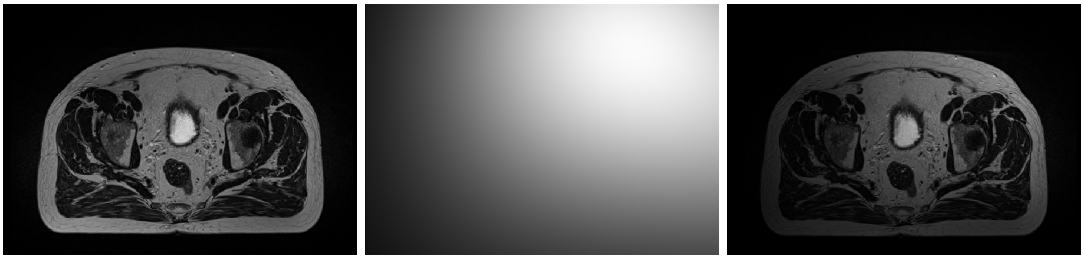
[1]Original MR image courtesy of Gentle Radiotherapy



(a) Original image slice

(b) A randomized bias field

(c) Original image slice multiplied with bias field results in this augmented image.



(d) Shifted original image slice (containing no negative values).

(e) The same randomized bias field as in Figure 4.5b

(f) Result of shifting the original MR image to contain only non-negative values before multiplying with bias field.

Figure 4.5: Example of bias field augmentation of MR images[1].

multiplication of the mesh and the image volume does not match the expected outcome. From Figure 4.5b, the expectation is a bright upper right corner in the augmented image slice, not a bright left corner as in Figure 4.5c. Therefore, the original MR image was shifted to contain only non-negative values, then multiplied with the bias field map, bf_{map}, and later shifted back as follows by

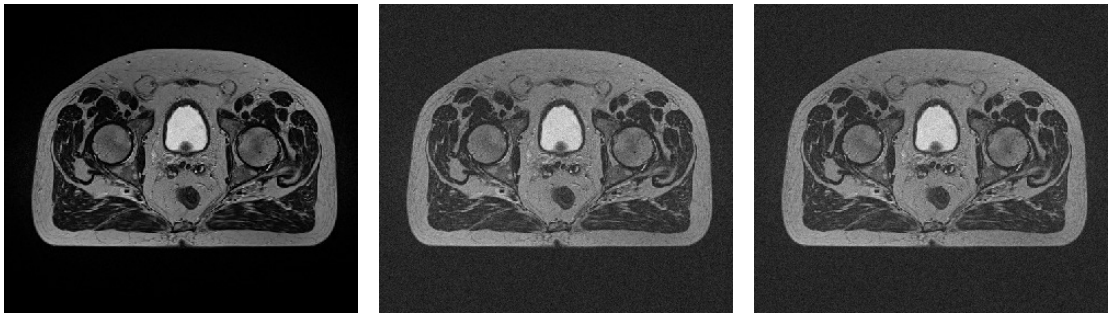
$$x_{aug} = (x + |x_{min}|) * bf_{map} - |x_{min}|, \tag{4.4}$$

where x_{min} is the minimum value in the original MR image, x. The result is shown in Figure 4.5f and matches the expected result better than in Figure 4.5c. Both approaches have been used in this project.

4.5.4 Augmentation by adding noise

A layer for adding noise was implemented, where each voxel value is modified by adding a noise value. The noise is user-specified as either Gaussian or Rician with customizable mean and standard deviation. Furthermore, the user can select which labels noise is added to, and individual parameters for different labels. Noise is added to the padding by default, but can be excluded if desired. Figure 4.6 shows an original MR image slice next to two image slices which

[1]Original MR image courtesy of Gentle Radiotherapy



(a) Original image slice

(b) Gaussian noise with mean 0.87 and standard deviation of 0.2 added.

(c) Rician noise with mean 0.87 and standard deviation of 0.2 added.

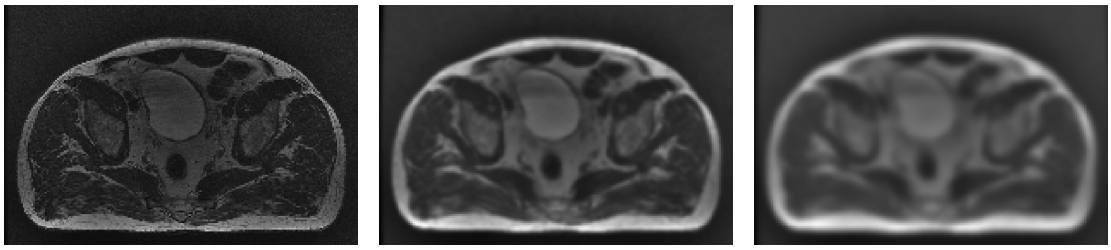
Figure 4.6: Example of augmentations by adding noise in MR images[1].

have been augmented with Gaussian and Rician noise. It is hard to visually distinguish between the two augmented image slices.

To introduce a wider variety during training, the layer was extended with an additional hyperparameter. This hyperparameter lets the user choose an interval for a uniform distribution from which the mean value is drawn. For each augmentation step, a new random mean is generated and used.

4.5.5 Augmentation by smoothing

A layer which smooths image volumes with a Gaussian kernel was implemented. The image volume is convolved with the kernel to produce an augmented volume. Since a Gaussian kernel was used, the voxels closest to the current voxel had a larger influence on the resulting value. The width (standard deviation) of the kernel determines the degree of smoothing, as illustrated in Figure 4.7. To create varying augmentation levels, an interval is specified by the user from which the standard deviation is drawn each time this augmentation is applied.



(a) Original image slice.

(b) Smoothing kernel with width 1.

(c) Smoothing kernel with width 2.

Figure 4.7: Example of augmentation by Gaussian smoothing in MR images[1].

[1]Original MR image courtesy of Gentle Radiotherapy

Table 4.3: Augmentation parameters for case 2.1 and case 2.2. For the flipping axes, l-r is the left-right axis and a-p is the anterior-posterior axis, i.e., front-to-back.

	Case	
Parameters	**2.1**	**2.2**
Scaling percentage interval	(-10.0, 10.0)	(-10.0, 10.0)
Flipping axes	l-r, a-p	l-r, a-p
Rotation angle interval	(-10.0, 10.0)	(-10.0, 10.0)
Elastic deformation, standard deviation	18	18
Elastic deformation, no. of control points	3	3
Gaussian smoothing, kernel width interval	(0.0, 1.0)	-
Noise, distribution	Gaussian	Gaussian
Noise, mean interval	(-1.5, 1.5)	(-1.5, 1.5)
Noise, standard deviation	0.2	0.2
Noise, labels	All	All
Bias field range	(-0.9, 0.9)	(-0.8, 0.8)
Bias field order	3	3
Bias field version	Original	Input image shifted
Histogram modification factor	1.5	1

4.5.6 Augmentation parameters

The order of the augmentation layers was histogram modification, followed by Gaussian smoothing, bias field, noise, flipping, scaling, rotation, and lastly elastic deformation. The different parameters used for case 2.1 and 2.2 are presented in Table 4.3. All augmentation layers had an independent probability of 50% of being applied, except for the elastic deformation, which had a 60% probability. The parameters were tuned in the same way as the other hyperparameters, i.e., by testing different values during development and selecting values according to validation performance. This resulted in the parameter values being the same for both cases, except for Gaussian smoothing (only applied to case 2.1), bias field, and histogram modification.

Chapter 5

Results

This section will present quantitative and qualitative results for cases 1 - 3, including a comparison between the results. The results presented here are the best out of the five repeated runs for each case. More details and results for all five runs on each case can be found in Appendices C - E. Results are presented with the mean DSC of all image volumes in the specified set, along with standard deviation and minimum values across the set.

5.1 Case 1 - Single site, no augmentation

As seen in Table 5.1, training on site A (case 1.1) resulted in a DSC of 0.886 with a standard deviation of 0.0615 on the validation set. The mean values between sites A - D cannot be said to differ. However, site D has a high standard deviation, 0.160 as compared to around 0.05 for sites A - C. The minimum DSC is higher for site A than for sites not included in training.

When training on site C (case 1.2), the result was a DSC of 0.668 with a standard deviation of 0.0771, see Table 5.1. Here, the validation score for site C is noticeably higher than for site A, B, and D, both in mean and minimum DSC. Site D had a standard deviation of 0.110, which was higher than for the remaining sites (< 0.065). Note that there were only two validation examples from site C for this case.

Table 5.1: DSC for training, validation, and test sets for case 1.1 and 1.2 together with the validation result for each site. The mean and minimum values as well as standard deviation for the DSC across the data sets are presented. N is the number of image volumes in each set.

	Case 1.1				**Case 1.2**			
Set	**N**	**Mean**	**SD**	**Min**	**N**	**Mean**	**SD**	**Min**
Training	40	0.97692	0.00118	0.97390	40	0.97095	0.00154	0.96668
Validation	90	0.88574	0.06153	0.54465	90	0.66774	0.07708	0.44111
A	13	0.90728	0.04986	0.79755	43	0.66484	0.06470	0.44111
B	37	0.89627	0.05295	0.66860	41	0.66327	0.06145	0.44353
C	35	0.87668	0.04613	0.72901	2	0.94148	0.00033	0.94125
D	5	0.82302	0.16013	0.54465	4	0.60790	0.11026	0.45034
Test	21	0.90112	0.03630	0.80073	21	0.66651	0.06581	0.55388

5.2 Case 2 - Single site, with augmentation

For case 2.1, a DSC of 0.927 was achieved on the validation set, with a standard deviation of 0.0195, as seen in Table 5.2. Comparing the mean validation scores for non training sites (B - D), they cannot be said to differ from the DSC of the training site (A). The standard deviation is higher for site D than for the remaining sites. Note that site D has a smaller sample of validation examples compared to site A - C.

The validation score for case 2.2 was 0.919 with a standard deviation of 0.0243, see Table 5.2. The mean DSC of the validation set for each site cannot be said to differ, although the minimum DSC for site C, which was used for training, is higher than for sites A, B, and D. The standard deviation is low for site C, and it should be noted that there were only two image volumes in the validation set for this site.

Table 5.2: DSC for training, validation, and test sets for case 2.1 and 2.2 together with the validation result for each site. The mean and minimum values as well as standard deviation for the DSC across the data sets are presented. N is the number of image volumes in each set.

	Case 2.1				**Case 2.2**			
Set	**N**	**Mean**	**SD**	**Min**	**N**	**Mean**	**SD**	**Min**
Training	40	0.94655	0.00636	0.93035	40	0.95191	0.00494	0.94069
Validation	90	0.92673	0.01950	0.83620	90	0.91904	0.02428	0.80562
A	13	0.93440	0.01532	0.90346	43	0.92050	0.02169	0.81321
B	37	0.92209	0.02212	0.83620	41	0.91719	0.02692	0.80562
C	35	0.93104	0.01115	0.89951	2	0.94064	0.00706	0.93565
D	5	0.90747	0.03829	0.84024	4	0.91143	0.02751	0.87328
Test	21	0.93013	0.01116	0.91175	21	0.91985	0.01611	0.87025

5.3 Case 3 - Multi-site

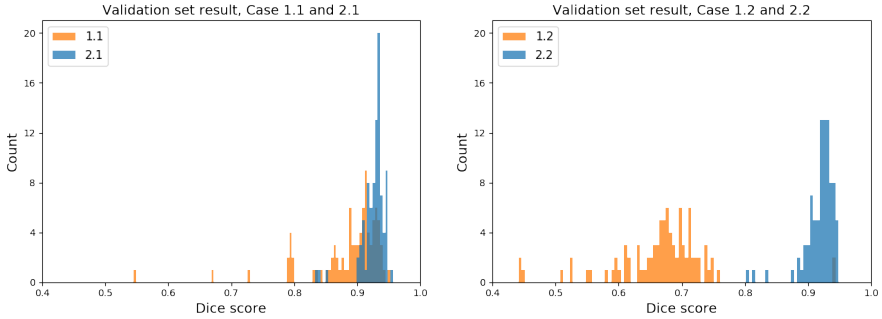
Training on the large multi-site data set (100 image stacks) resulted in a DSC of 0.930 and a standard deviation of 0.0206, see Table 5.3. The results are similar for sites A - C ($\approx$0.93), whereas site D achieved a lower score of 0.880 and a rather high standard deviation of 0.0564. It should be noted that there are only two image stacks in the validation set for site D. Site A has a very low standard deviation of 0.00815 for its 11 validation image volumes.

Table 5.3: DSC for training, validation, and test sets for case 3 together with the validation result for each site. The mean and minimum values as well as standard deviation for the DSC across the data sets are presented. N is the number of image volumes in each set.

	Case 3			
Set	**N**	**Mean**	**SD**	**Min**
Training	100	0.96500	0.00269	0.95645
Validation	36	0.93002	0.02063	0.84008
A	11	0.93670	0.00815	0.92365
B	11	0.93326	0.01716	0.88883
C	12	0.92928	0.01500	0.89432
D	2	0.87995	0.05638	0.84008
Test	15	0.93124	0.01450	0.91262

5.4 A discussional comparison

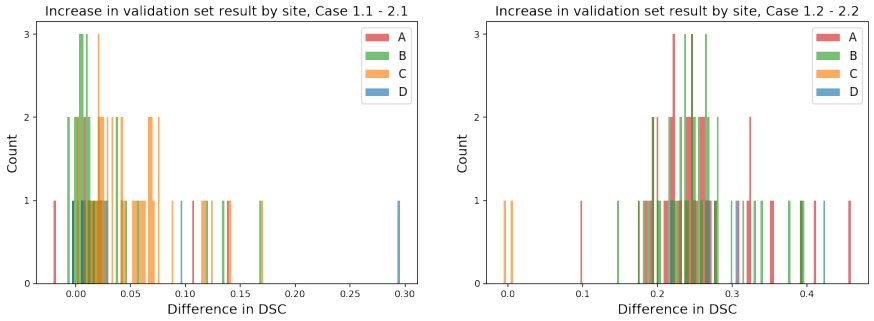
Using the augmentation techniques presented in Section 4.5, the performance increased from case 1 to case 2. For case 2.1, which had the exact same data split as case 1.1, the results increased from 0.886 to 0.927, and the minimum value in the validation set increased from 0.545 to 0.836. Although the performance between sites could not be said to differ in case 1.1, the total validation results increased for case 2.1, which can be seen in Figure 5.2a. For case 2.1, the DSC peak is more narrow and located around higher scores than for case 1.1. Figure 5.1a shows the difference in DSC for each validation subject between case 1.1 and 2.1, from which it can be observed that augmentation had a larger effect on site C than on site B.



(a) Case 1.1 and case 2.1

(b) Case 1.2 and case 2.2

Figure 5.1: Histogram over the results on 90 validation image volumes for case 1.1 and 1.2 in comparison with case 2.1 and 2.2.

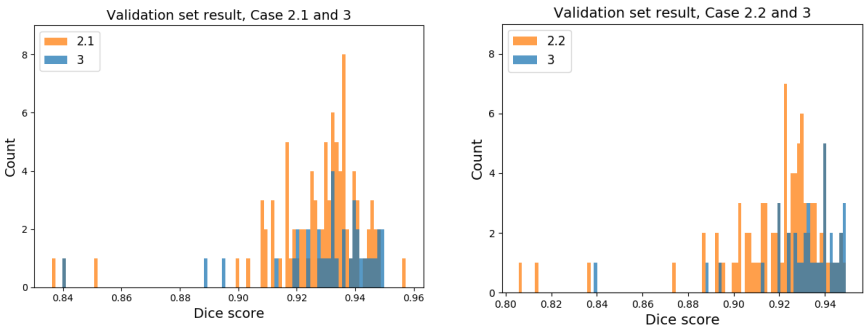


(a) Change from case 1.1 to case 2.1

(b) Change from case 1.2 to case 2.2

Figure 5.2: Histogram over the difference in DSCs for each image volume in the validation set by site when adding augmentation.

Training on site C with augmentation increased the result from 0.668 to 0.918, and the minimum value increased from 0.441 to 0.810, from case 1.2 to 2.2, respectively. The differences between site C and non-training sites evened out, as all sites had similar DSCs when using augmentation. There was no difference in the result on the image stacks from site C, neither better nor worse. Figure 5.2b shows the difference in DSC for each validation image stack between case 1.2 and 2.2 which illustrates the large shift of DSCs when applying augmentation.



(a) Case 2.1 and case 3 **(b)** Case 2.2 and case 3

Figure 5.3: Histogram over the results for case 2 and 3 on 90 and 36 validation image volumes, respectively.

The results when training on 40 image volumes with augmentation (case 2.1 and 2.2) were similar to the results which were acquired when training on 100 volumes (case 3), which can be seen in the inseparable histograms in Figures 5.3. It should be noted that case 3 only has 36 validation image volumes, while case 2.1 and 2.2 have 90.

The results for case 1 - 3 are also visualized in the box plot in Figure 5.4. The box representing the $25^{th} - 75^{th}$ percentile is more narrow for case 2 and 3, as compared to case 1. Especially case 1.2 has a large range from lowest to highest DSC. The lowest values for the cases with augmentation and large training data set are all above the minimum whisker for case 1.1 and the maximum whisker for case 1.2.

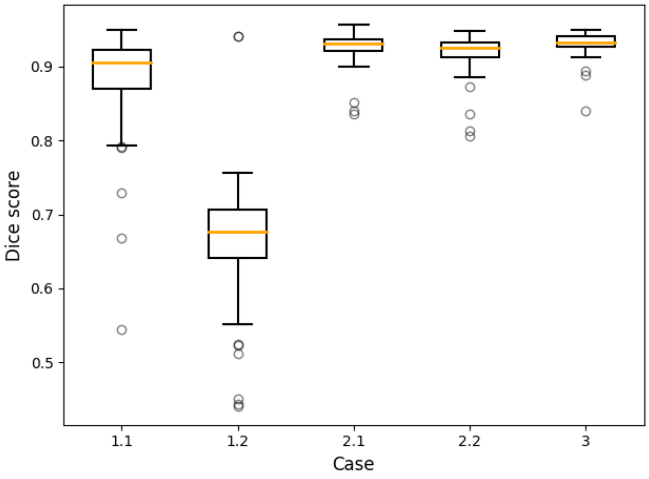


Figure 5.4: Boxplot over DSC in the validation set for each case.

Analyzing how the network models differ in performance considering the different organs show that the network in case 1.1 was fairly good at all organs, but had low minimum values for especially bladder, rectum, and prostate. The mean values for these organs increased by $0.0355 - 0.0816$ and the minimum values by $0.178 - 0.671$ when adding augmentation. The network in case 1.2 did not perform well on any labels with poor minimum values as well.

Table 5.4: Mean and minimum DSC for the different organs on the validation sets. Femoral heads are here abbreviated as FH.

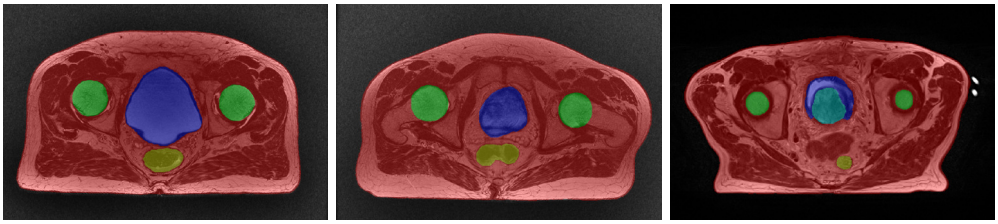
	Mean DSC					Min DSC				
Organ	**Case 1.1**	**Case 1.2**	**Case 2.1**	**Case 2.2**	**Case 3**	**Case 1.1**	**Case 1.2**	**Case 2.1**	**Case 2.2**	**Case 3**
Body	0.98934	0.94290	0.99087	0.98727	0.99339	0.95593	0.76290	0.98121	0.96620	0.98823
FH	0.94390	0.87285	0.94990	0.94939	0.95567	0.83758	0.69172	0.92146	0.90469	0.93727
Bladder	0.90514	0.07838	0.94067	0.93008	0.95332	0.31950	0.00132	0.49706	0.39506	0.86785
Rectum	0.79368	0.77633	0.87401	0.85958	0.87238	0.31357	0.27534	0.69925	0.64308	0.71562
Prostate	0.79662	0.66823	0.87822	0.86886	0.87537	0.10740	0.00060	0.77878	0.73801	0.67421

All mean and minimum values increased by adding augmentation, see Table 5.4. For bladder, rectum, and prostate, the increase was by $0.0833 - 0.852$ and $0.368 - 0.737$ for the mean and minimum values, respectively. The maximum values and standard deviations are presented in Appendix G.

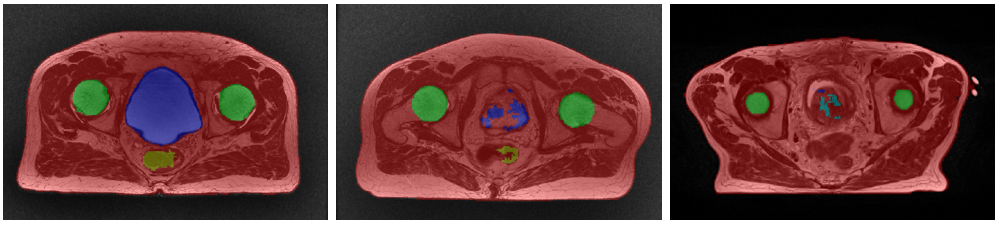
Case 3 had higher minimum values for both bladder and rectum than both versions of case 2. The subject with the lowest DSC on bladder for case 2.1 and 2.2 is the same example. This image stack is not in the validation set for case 3, but in the training set instead. It is interesting to note that the minimum value on the prostate is lower on case 3 than on both versions of case 2. The image volume with this score was also a validation example in case 2.1 and 2.2.

Case 1.1, which was trained on site A had a validation score on site A of 0.907. Case 2.2, on the other hand, had a DSC of 0.921 on the validation set from site A, even though only trained on site C.

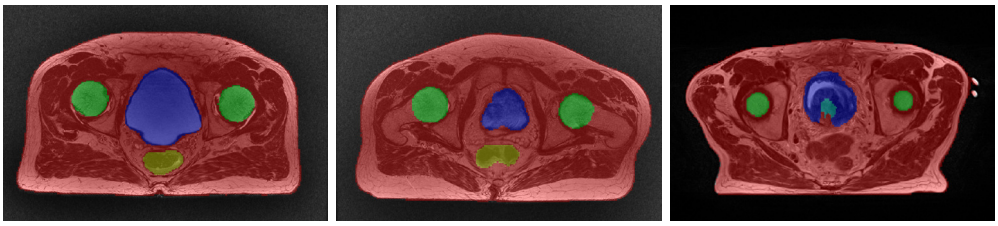
When training on site C, as in case 1.2, the validation set of site C had a DSC of 0.941. Case 2.1, trained on site A, had a DSC on the validation set of 0.931. In case 1.2, there is only two image stacks from site C in the validation set whereas there are 35 site C examples in the validation set in case 1.2.



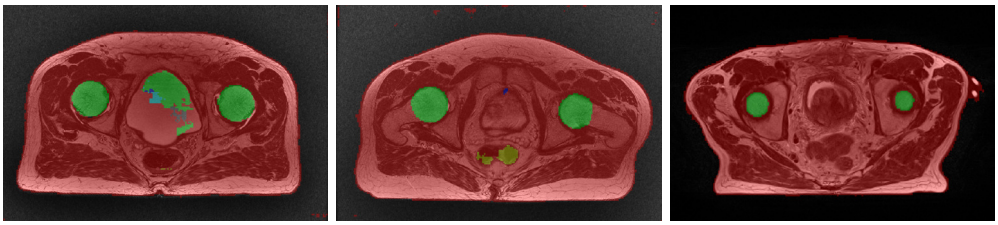
(a) Ground truth **(b)** Ground truth **(c)** Ground truth



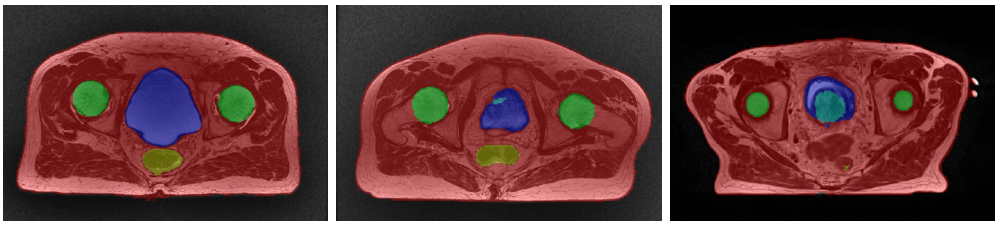
(d) Case 1.1, DSC=0.7975 **(e)** Case 1.1, DSC=0.6686 **(f)** Case 1.1, DSC=0.5446



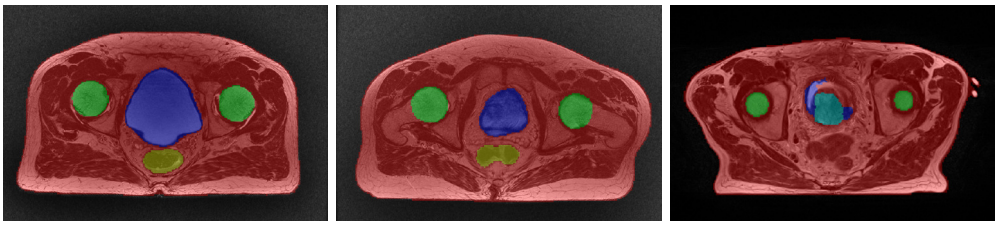
(g) Case 2.1, DSC=0.9357 **(h)** Case 2.1, DSC=0.8362 **(i)** Case 2.1, DSC=0.8402



(j) Case 1.2, DSC=0.4411 **(k)** Case 1.2, DSC=0.6113 **(l)** Case 1.2, DSC=0.4503



(m) Case 2.2, DSC=0.9000 **(n)** Case 2.2, DSC=0.8056 **(o)** Case 2.2, DSC=0.8733



(p) Case 3, DSC=0.9472 (training) **(q)** Case 3, DSC=0.9565 (training) **(r)** Case 3, DSC=0.8401

Figure 5.5: Comparison of segmentations for three different patients from site B, C, and D, respectively. The MR images are from the validation set if not otherwise stated. Original MR images courtesy of Gentle Radiotherapy.

5.5 Illustrative examples

To further investigate the effect of augmentation, a qualitative comparison is presented in this section. The qualitative performance increased as well, which is illustrated in Figure 5.5 where ground truths are presented along with segmentations from all cases for three different validation examples. The same comparison, but for the sagittal plane can be found in Appendix F.

For the image in the left column in Figure 5.5, case 1.1 seemed to perform well, although the rectum was better delineated in case 2.1 and DSC increased with 0.138. Case 1.2 was not able to handle this image and failed to segment bladder and rectum. With augmentation (case 2.2), the performance increased greatly which is seen in the figure and proven by an increase of 0.459 in DSC. This image was in the validation set for all cases except case 3. Even so, case 2.1 achieves almost the same DSC as case 3.

For the image in the middle in Figure 5.5, the networks in both case 1.1 and 1.2 failed to segment bladder and rectum. With augmentation, the DSC in case 2.1 and 2.2 increased by 0.168 and 0.194 on the whole image volume, respectively. The results can still be improved further since a small part of the bladder was segmented as prostate for case 2.2, and a small part of the bladder is missing in all cases except for case 3, where the image was a training image.

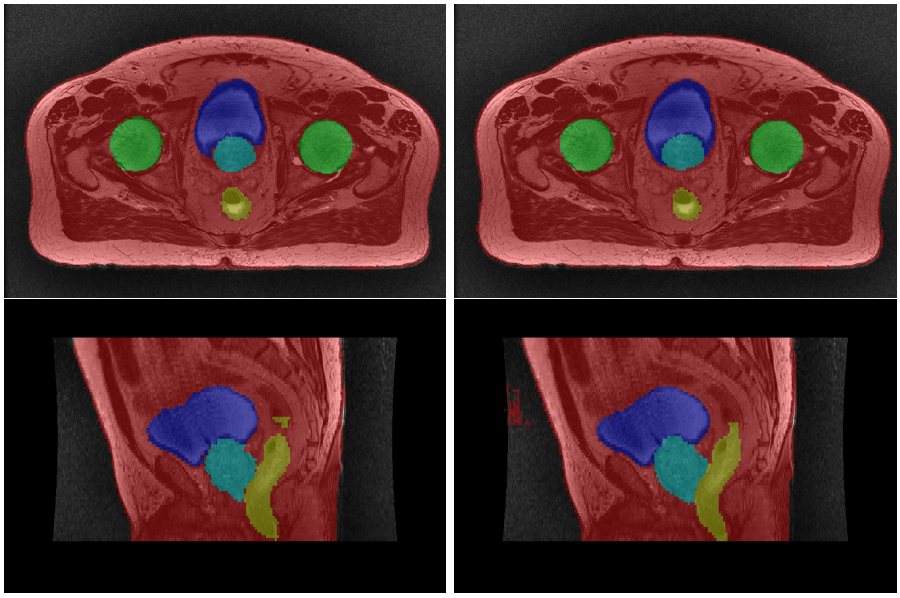
The rightmost image in Figure 5.5 was troublesome for the networks in case 1.1 and 1.2 regarding bladder, prostate, and rectum. Performance increased with augmentations in both case 2.1 and 2.2. From case 1.1 to 2.1, the increase in DSC for this image stack was 0.296, and from case 1.2 to 2.2 an increase of 0.398. This image stack has been consistently causing problems for all trained networks, including case 3. No network succeeded fully in excluding the catheters (the bright white spots on the right side of the image) from the segmentation.

The best validation example from case 1.1 had a DSC of 0.949. This image volume is from site A, which case 1.1 was trained on. Comparing this with case 2.2, which did not have any site A images in training, see Figure 5.6, shows that case 2.2 handles it even better, proven by a higher DSC. However, there is a noticeable error in the body segment, which is wrongly placed in the air on the left side of the sagittal slice.

In the same way, the best validation example from case 1.2, with a DSC of 0.942, was compared to the segmentation produced by the network in case 2.1, which had a DSC of 0.941. This image is from site C, which was the training site in case 1.2, but not in case 2.1. The two segmentations are shown in Figure 5.7, where the organs seem to be handled well, but there is some loss in the body segment, which can be seen from both the transverse and sagittal slices. Notice that case 1.2 does not handle the body completely either, as there are some small holes in the segment, as can be seen in the transverse slice.

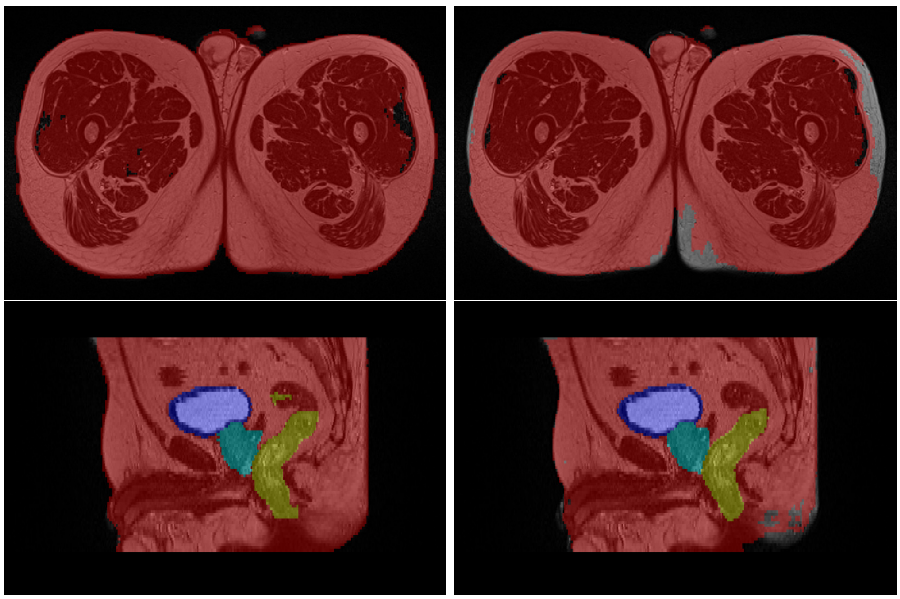
5.6 Repeatability

The performance of each network varied between runs. For case 1.1, the validation set performance varied between 0.765 - 0.886, for case 1.2 between 0.562 - 0.668, for case 2.1 between 0.910 - 0.927, for case 2.2 between 0.912 - 0.919, and lastly, between 0.926 - 0.930 for case 3. The complete tables with results can be found in Appendices C - E.



(a) Case 1.1, DSC = 0.9491 **(b)** Case 2.2, DSC = 0.9489

Figure 5.6: The best validation example from case 1.1 (trained on site A) in comparison with case 2.2 (trained on site C). The validation example is from site A. [1]



(a) Case 1.2, DSC = 0.9417 **(b)** Case 2.1, DSC = 0.9412

Figure 5.7: The best validation example from case 1.2 (trained on site C) in comparison with case 2.1 (trained on site A)[1]*. The validation example is from site C.*

[1]Original MR images courtesy of Gentle Radiotherapy

Chapter 6

Discussion

In this thesis, the aim was to investigate the potential improvement in robustness for multi-site segmentation by using augmentation. When training a network on a single site, it was found that using augmentation improved results on the multi-site validation set. To further study the effect of augmentation, the performance on individual sites and sites with different vendors were analyzed for two different sites in training. The results were also compared to the performance of a network trained on a multi-site data set.

In this section, the results will be discussed together with possibilities and restrictions imposed by the manual segmentations, hardware, methods, and data set. Lastly, ideas for further work are presented.

6.1 Creation of ground truth

As expert segmentations were not available as ground truth, the results in this report do not indicate an exact number for delineation performance. Instead, it can be seen as an indication for the possibilities for automatic segmentation of target and OARs and the effect of augmentation.

One limitation in the prostate segmentation approach is that the seminal vesicles have not been included. The reason for this was the lack of decision basis as the patient journals have not been available during segmentation. Therefore, the prostate segmentation provided by the networks in this report can be seen as a starting point, which might need further adjustments if for example, the seminal vesicles should be included.

As described in Section 2.4.2, experts show a variation of up to over 15% for rectum. The presented variability is most likely due to uncertainties or different standards on how much of the rectum should be segmented, i.e., how many transverse slices should be delineated. Thus, the same organ volume might be segmented, but in different image slices, resulting in a lower DSC, since the measure only considers exact segment overlap. In this thesis, the results from the inter-expert variability study have been viewed as the human level performance the network should aim for (or surpass).

Presumably, there is a variability between the segmentations in this thesis as well. The lack of prior segmentation experience in combination with the variability between segmentations has most likely created an upper limit for network performance with implications for the final segmentation result.

6.2 Limitations posed by hardware

Since CNNs are computationally expensive, the available hardware in combination with the chosen network posed restrictions on the training settings. For example, in this thesis, at least 22 Gb GPU was required to allow weight updates based on two image volumes instead of a single volume. This was the case for all computers except one. The performance increased considerably when using a batch-size of two as compared to a batch-size of one. A larger batch-size could potentially improve the results further, but demands more computational resources than were accessible during this work. The results presented for cases 1 - 3 were therefore all run with a batch-size of two. However, the side projects presented in Appendix H were run with a batch-size of one, thus not directly comparable to the main cases.

The restrictions posed by the GPU resulted in a trade-off for the hyperparameters for batch-size, resolution, and window sample size. To keep a batch-size of two and include a rather large volume of the MR image stack, the images had to be downsampled, which in turn led to the resolution of the resulting segmentations: $2 \times 2 \times 2.5$ mm. This might be considered a bit too sparse in resolution, as the output segmentation edges were not that smooth. This could be mitigated by postprocessing the segmentations and result in more visually pleasing segmentations, although not necessarily generating a higher DSC.

6.3 Dice as evaluation metric

When evaluating the performance of the network, DSC is widely used in medical segmentation applications [26] and was believed to be a good approximation of segmentation quality for this work. As long as a higher average DSC for the validation set represented better overall segmentations, there was no need for another metric. However, as DSC only measures exact overlap, this brings limitations when evaluating individual results as the measure does not consider the location of a misclassification. For example, placing a rectum segment of n pixels in the background (air) has the same effect on the DSC as delineating rectum one extra slice of size n pixels in the transverse plane, even though the latter indicates a better network model.

DSC can also be misleading when segmenting multiple structures with varying sizes. In this work, the body label is much larger relative to the OARs, resulting in high DSC even though there are apparent misclassifications of large segments. Figure 5.7 illustrates this, where the left body segment had a DSC of 0.995 and the right had a DSC of 0.992. The difference in DSC is small, even though there are apparent differences between the two.

6.4 Hyperparameter tuning

One challenge during the course of this thesis was to optimize hyperparameters. Even though it was not the main focus to fully optimize the network, it was desirable to optimize each network in order to compare the results on equal grounds. Training a network for 30 000 iterations and a batch-size of two took between 19 and 38 hours, depending on the graphics card properties. The long training time ruled out optimization algorithms. The results from the repeatability study challenged the usefulness of grid-searches since an evaluation of a changed hyperparameter value did not necessarily mean that the parameter was responsible for the difference in DSC. The difference could also have been due to randomness, which is further discussed in Section 6.5.4.

A full configuration of a network includes many more hyperparameters to tune than have been explored in this work. For example, the network and optimizer were fixed to decrease the amount of parameters to tune. One issue that has been discussed throughout this work is if the

results are comparable as it is not known if the settings are fully optimized for each individual network. However, as the tuning of case 1 - 3 regarded the same parameters and the training performance converged for all cases, the relative result is not believed to differ considerably. The large difference between validation and training set performance for case 1.2 (0.67 and 0.97) was not believed to be due to overfitting. This was confirmed by visualization of the loss function throughout training and that the model did not generalize better to the validation set at any other stage during the 30 000 training iterations.

6.5 Augmentation improves robustness

In case 1.2 the non-training site DSCs ($0.608 - 0.665$) compared to the training site DSC (0.941) shows that varying characteristics are a problem for CNNs trained on a single site, regardless of vendor. From case 1.2 to 2.2, the mean validation DSC increased $0.254 - 0.304$ for non-training sites, confirming improved robustness using augmentation.

The extent of the problem with single site training depends on the training site characteristics. For case 1.1, the difference between training (0.907) and non-training site ($0.823 - 0.896$) DSCs were not that large to begin with. One reason could be that the training image volumes were noisier than for case 1.2, indicating that it is easier to learn from images with lower SNR and infer to images with higher SNR than vice versa. The increase in robustness was not as prominent for non-training sites from case 1.1 to 2.1, $0.0258 - 0.0845$, although the minimum DSC increased by 0.292.

The achieved mean values were within a 0.011 range when training on 40 image stacks from a single site with augmentation and when training on 100 image stacks from all sites. As illustrated in Figure 5.3 in the Results section, the distributions of the validation DSCs are not expected to differ. This shows that augmentation can alleviate the need for a large data set when training CNNs.

The side project presented in Appendix H.2, where augmentation was added during inference, strengthens the claim that augmentation makes the network robust. Even though the image stacks in inference were distorted in a similar way to the image stacks in training, thus resembling the training set characteristics, equivalent results were obtained. Hence, the network produces segmentations with the same DSCs regardless if it is given original or distorted image stacks.

All analyzes were made on the validation sets as these data sets were larger. However, the DSCs were equivalent to the ones for the test sets for all cases, so the same conclusions are expected. The test set results were used as a final check of model performance, but have not been evaluated further. The large validation sets are thought to have been a contributing factor to the non-existent problem of overfitting to the validation set.

6.5.1 Augmentation does not need to mimic reality

In the beginning of this work, augmentation was applied in order to mimic other sites and produce as realistic MR images as possible. However, as the network should perform well on image volumes from all sites, also if presented with an image volume from a new site with unknown properties, as large variation as possible was desired. Consequently, the augmentation techniques were used for exaggerating properties instead. The exaggeration technique created distorted image volumes which challenged the network to find useful features and led to higher DSCs for the validation sets. For example, the image in Figure 4.4c does not resemble an original MR image, but has a bladder with the same intensity as the rest of the organs. One proposed reason why this improved performance is that the network can then learn that the bladder wall is a useful landmark, rather than learning that "the bright(est) area" represents the bladder.

Based on the initial approach, both introducing and reducing noise was of interest for investigation. Simulation of noise-free image stacks was attempted by smoothing images with a Gaussian kernel. However, this did not turn out as hoped. Augmentation by smoothing seemed to decrease performance for case 2.2 and was therefore excluded, but kept in case 2.1 as no reduction of DSC was noticed. The effects of the smoothing augmentation should be further investigated before a conclusion of its effects can be drawn.

Adding Rician noise to the image stacks did not improve segmentation results. This is another indication to not mimic what is seen or known in theory when augmenting, but rather focus on exposing the network for variation during training.

The online augmentation method enabled the large augmentation variation during training, and was implemented as layers in NiftyNet. However, there are more advanced methods for generating image stacks, for example by using generative adversarial networks (GANs) [57, 58]. A GAN needs to be trained, and would for each input image stack produce an output with a specific style or characteristic. The same GAN will probably generate similar characteristics for all input image stacks. Our method does not necessarily assume anything about which sites the network should be used on, and still performs well. We believe that our chosen method provides more flexibility and can introduce a larger variation than by using GANs.

The augmentation approaches were selected based on general MR image differences and variations due to acquisition protocols, such as varying noise levels, bias fields, and histogram differences. As far as we know, augmentation by histogram modification has not been used before, even though preprocessing steps including histogram equalization or normalization are common. Geometric augmentations are widely used and often believed to enhance performance [59]. In this work, elastic deformation introduced a variability that mitigated problems with deviant patient geometries, proven by an increase in minimum DSC for each site. This is further discussed in Section 6.5.3.

6.5.2 Data set site imbalance

The used data set was remarkably large considering the field of investigation. The data set allowed a large set of image volumes throughout the training, validation and test sets. However, the distribution between the sites was skewed as one site only had six image volumes while another had more than 40. The amount of image volumes from GE (103) was also twice as many as from Siemens (48). These differences affect the between-site and between-vendor comparisons.

The small number of validation examples from site C in case 2.2 and from site D in all cases makes it hard to draw conclusions when comparing results. However, the results for site C are merely regarded as an indication, and given more image stacks for site C in validation, the results are not thought to deviate a lot from the presented DSCs. This is because in case 2.1, the standard deviation on the validation DSC for site C (0.0112) was low as well as the standard deviation of the validation set for image stacks from the training site (0.0153). There is no reason to believe that the network trained on site C would have a larger standard deviation for site C than for a network not trained on that site.

On the other hand, there are only six image stacks from site D in total, where one has been difficult to segment both during manual segmentation, and for all networks. The performance on this example can therefore have a big influence on the DSC for site D, both considering mean value, minimum value, and standard deviation. One might argue that site D should have been excluded from the comparisons due to the few available image stacks and difficult geometries. However, an automatic segmentation tool has to perform on *all* image stacks, since all patients need a delineation as basis for dose planning.

6.5.3 Outliers

From what has been observed in this work, elastic deformation was highly effective in increasing the performance of "outliers", i.e., patients with deviant anatomy. This has been recognized due to the increase of minimum DSC in validation and test sets for networks were augmentation was included in training. During discussions on how to improve the network performance further it has been suggested to exclude outliers from the training set. This can increase performance on "normal" image volumes, but outliers in the training set most likely contribute with a similar effect to that of elastic deformation, as they introduce anatomy variations.

When observing the final results for case 2, the difficulties seem to be regarding anatomical differences rather than site differences. There is also a possibility that the ground truth segmentation variability is restricting further improvement, and not the network. Outliers were not difficult only for the network, but during manual segmentation as well. In Appendix F, Figure F.1c, Figure F.1i, and Figure F.1o, presents ground truth and network segmentations (case 2), where the network results look more reasonable in the sagittal plane. However, that does not guarantee a more correct overall segmentation.

6.5.4 Implications of repeatability results

The repeatability investigation showed that for networks which performed badly on the validation and test sets, the variation between different runs were larger. The network most likely finds different sets of features during training, which generalize more or less to the validation set. However, it still fails to perform well in all repeated trainings. A smaller variation was observed for the well-performing networks (case 2 - 3), which indicates that augmentation, or a large training set, helps the network to find more robust features. The model with the best validation performance was selected as the final model. In all cases, this model also achieved the highest DSC on the test set, which validates the choice. This indicates that it could be a good idea to train a network multiple times and select the best one in order to improve model performance. However, since the between-run variation is still present it is likely that complete convergence is not reached for the trained models. This variation could be due to the choice of optimizer (Adam) as it has been shown to not generalize as well as SGD in all cases [20].

6.5.5 Generalizability

The augmentation methods that were investigated and developed during this thesis have been applied on an MRI case study. We believe that, especially when dealing with data from different sites with different characteristics, introducing variation in the training set will make the network more robust and invariant to varying characteristics. Therefore, even though for example the bias field augmentation is specialized for MRI, it could potentially be useful for other modalities as it introduces variations which could make a CNN more robust.

6.5.6 Clinical relevance

The aim of augmentation was to improve robustness for multi-site segmentation, which has been evaluated on four sites with machines from two different vendors. The final network models were also evaluated on MR images from a fifth site with a third camera vendor. The result looked promising, with better performance for the augmentation networks than for those trained without, but could not be quantitatively evaluated due to lack of ground truth. The qualitative result cannot be published as no publishing rights have been granted for these images.

Throughout the thesis, we have had in mind that the ultimate goal for automatic segmentation is not a perfect DSC, but rather a segmentation that a clinician can accept without further modifications and which does not affect clinical outcome, i.e., the radiation dose to the patient. It would be interesting, though not possible at this stage, to evaluate the clinical effect of using the proposed segmentation networks without any modification. In that case, it would be necessary to generate dose matrices for both a manual segmentation and these network segmentations, and compare them.

The achieved results are comparable to the inter-expert variability presented in Section 2.4.2, with better numbers for especially femur and rectum delineations, which could be due to lack of consensus for these organs in the Gold Atlas that was used. The mean value for bladder was better for case 2 and 3, but the minimum values were worse for case 2 than in the study. The study lacked prostate segmentation variabilities.

Prostate segmentation has been an interesting topic for research, and there is a global challenge, called PROMISE12 [60], where teams are given 50 multi-site MR images and segmentations of the prostate for training and 30 image volumes for test, in order to create automatic segmentation algorithms. The participating teams are evaluated on four different metrics, where DSC is one of them. The challenge has been going on for seven years, and the current leader achieved a mean DSC of 0.870 on the whole prostate and a minimum value of 0.759 on April 1, 2019. However, it should be noted that there might be teams with higher DSCs, but worse on other scores, thereby not in first place. Also, the MR images in PROMISE12 do not have as large FOV as the images used in this project, but the DSC is still a good indication of what other studies have achieved. The mean and minimum DSC in this thesis were 0.878 and 0.779 for case 2.1, as well as 0.869 and 0.738 for case 2.2.

Automatic segmentations have the potential to save valuable time. In the process of establishing ground truth segmentations, around 30 minutes was spent on each patient. Although, it should be noted that the sCT was thresholded to speed up the segmentation of the body by having an initial segmentation as a starting point. With the trained CNNs in this thesis, a segmentation can be produced in less than one minute. In addition to evaluating clinical outcome of using the network segmentations directly, it would also be valuable to evaluate the time spent by a clinician to modify the segmentations.

6.6 Suggestions for related research topics

During this work, insight has been gained regarding augmentation techniques and medical image segmentation. There are many interesting topics which could be subject to further research, which are briefly explained in this section.

A study which maps the contribution of each augmentation technique to the results, also referred to as an ablation study, would be interesting. Since the repeatability for the network training showed that results varied a lot, it was difficult to gain knowledge in which hyperparameters contributed the most to an improved DSC. There are possible ways to reduce the variability for the sake of such a study, for example could using log files or seeds for the random number generators be helpful.

To improve segmentation results further, it would be interesting to investigate ways of post-processing the network segmentations. For example, holes in the body segment should not be present, and an organ segment should always be coherent (except for the femoral heads which consists of two coherent segments).

The side project presented in Appendix H.1 showed that by using only four image stacks, one from each site, the results are decent. This opens up for questions regarding how many image stacks are really needed for training. Furthermore, if new sites are added, is it enough to

include just a few image stacks in training to perform well on that site? These questions could be of interest to follow up on.

Augmentation proved to work well and increase performance for the networks which performed badly from the beginning (case 1). It would be interesting to investigate the effect of augmentation on a network which already has good performance, such as case 3. As already discussed, patient anatomy is believed to be the biggest challenge for further improvements, hence, a suggestion is to investigate elastic deformation and other geometric augmentation techniques further to better deal with outliers.

On the topic of improving segmentation performance, the loss function can also be modified. The same reasoning as for DSC as evaluation metric in Section 6.3 can be applied to using Dice as loss function, i.e., Dice works initially but could be improved as results get better. The Wasserstein loss function, introduced by Fidon et al. [61] to handle imbalanced multi-class segmentations, could be of interest to investigate further. This loss function treats misclassified segments differently, here by a user-defined inter-class penalizing matrix. On the same topic, further studies could also include an investigation of evaluation metrics, e.g., distance-measures to distinguish between better or worse misclassifications. Other factors that may be desired to quantify is boundary smoothness. This can be important for the visual satisfaction of an automatic segmentation tool. The article by Taha and Hanbury [26] is recommended since a range of metrics for medical image segmentation are summarized and discussed.

Chapter 7

Conclusion

Automatic organ segmentations for multi-site MR images by using CNNs have been proven to be challenging due to site and vendor differences. In this thesis, a collection of augmentation techniques have been described and evaluated in a multi-site setting. The augmentations improved network robustness when training on a single site, proven by an increase in validation performance. The difference between training site and non-training sites decreased, in addition to an increased performance on difficult anatomies compared to when no augmentation was used. When augmentation was included in training, the results reached similar DSC to a CNN trained on a multi-site data set more than twice as large.

In our experience, introducing a large variety to the training set is beneficial. This can be achieved by online augmentation and using several augmentation techniques. For this thesis, bias field, noise, and histogram modification decreased differences in performance between sites, and elastic deformation increased performance on outliers. Other techniques include rotation, scaling, flipping, and smoothing, which have also contributed to the network robustness, but their individual effects have not been studied. The techniques were included or implemented in the NiftyNet workflow which is built on Tensorflow.

During this thesis, DSC was used as a single evaluation metric. However, even though DSC indicates the segmentation quality, it lacks possibility to measure and account for different types of misclassifications. Including measures that addresses this issue should be considered during further development of automatic organ segmentation.

An extension to the evaluation should consider including measures that addresses this issue.

The ground truth used when training the CNNs presumably contains a variability that imposes an upper limit for what DSC the network can achieve. However, the results indicate that human-level performance is achieved.

An automatic segmentation of target and OARs for prostate cancer can save valuable time for clinicians, and reduce present inter-expert variability. The time needed for a clinician to correct a network-produced delineation needs to be evaluated and considered in further development of an automatic segmentation tool.

Appendix A Hyperparameters for all cases

Parameters	Case					Side projects					
	1.1	1.2	2.1	2.2	3	Training on 4 images	Extreme aug	Organ net	Body net	Organ net	Aug in inference	Aug in inference
							No aug		With aug			
Iterations	30 000	30 000	30 000	30 000	30 000	15 000	20 000	20 000	20 000	30 000		
Learning rate	0.0100	0.0100	0.0010	0.0010	0.0010	0.0010	0.010	0.010	0.0010	0.0010		
Learning rate decay factor	1/8	-	1/3	1/3	1/3	1/3	-	-	-	1/3		
Learning rate decay iteration	20 000	-	10 000	10 000	10 000	5000	-	-	-	10 000		
L^2 reg term	0.00001	0.00001	0.00001	0.00001	0.00001	0.0001	0.00001	0.00001	0.00001	0.00001		
Normalization	False	True	False	False	False	False	True	True	False	False		
Dropout	0.0	0.1	0.0	0.0	0.0	0.01	0.1	0.1	0.0	0.1		
Scaling			(-10.0, 10.0)	(-10.0, 10.0)		(-15.0, 15.0)			(-10.0, 10.0)	(-10.0, 10.0)	-	-
percentage			l-r, a-p	l-r, a-p		l-r, a-p			l-r, a-p	l-r, a-p	-	-
Flipping axes			(-10.0, 10.0)	(-10.0, 10.0)		(-10.0, 10.0)			(-10.0, 10.0)	(-10.0, 10.0)	-	-
Rotation angle			18	18		15			15	15	-	-
Elastic deformation (sigma)			(0.0, 1.0)	-		(1.0, 2.0)			-	-	(0.0, 0.5)	-
Gaussian smoothing (sigma)			(-1.5, 1.5)	(-1.5, 1.5)		(-1.5, 1.5)			(-1.5, 1.5)	(-1.5, 1.5)	(-1.5, 1.5)	(-1.5, 1.5)
Gaussian noise mean (interval)			0.2	0.2		0.3			0.2	0.2	0.2	0.2
Gaussian noise sd			(-0.9, 0.9)	(-0.8, 0.8)		(-0.9,0.9)			(-0.8,0.8)	(-0.8,0.8)	(-0.3,0.3)	(-0.3,0.3)
Bias field range			1.5	1		1.5			1	1.5	0.5	0.33
Histogram augmentation factor												

Appendix B Histogram augmentation algorithm

Algorithm 1 Histogram-augmentation

function ScalePercentiles($perc, low\_limit, high\_limit$)
  $slope \leftarrow (high\_limit - low\_limit)/(perc[last] - perc[first])$
  $intercept \leftarrow low\_limit - slope \cdot perc[first]$
  $scaled\_perc \leftarrow slope \cdot perc + intercept$
  return $scaled\_perc$
end function

function ModifyPercentiles($image, factor$)
  $perc \leftarrow ComputePercentiles(image)$ ▷ Computes the following percentiles: [0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95]
  $perc \leftarrow ScalePercentiles(perc,\ 0,\ 100)$
  for all $p \in perc$ **do**
    $p \leftarrow p + random.normal(0, perc \cdot factor)$ ▷ Random number from Gaussian distribution with zero-mean and $sd = perc \cdot factor$
    if $p < 0$ **then**
      $p \leftarrow random.uniform(0, 1)$ ▷ Random number from uniform distribution in range [0,1)
    else if $p > 100$ **then**
      $p \leftarrow 99 + random.uniform(0, 1)$
    end if
  end for
  $perc \leftarrow sort(perc)$
  return $perc$
end function

function Transform($image, mapping$)
  $perc \leftarrow ComputePercentiles(image)$
  $diff\_mapping \leftarrow mapping[1 : last] - mapping[0 : last - 1]$
  $diff\_perc \leftarrow perc[1 : last] - perc[0 : last - 1]$
  $slopes \leftarrow diff\_mapping/diff\_perc$
  $intercepts \leftarrow mapping - slopes \cdot perc$
  for all $pixel \in image$ **do**
    $bin\_id \leftarrow bin(pixel)$ ▷ Find what percentile the pixel value belongs to
    $image[pixel] \leftarrow image[pixel] \cdot slopes[bin\_id] + intercepts[bin\_id]$
  end for
  return $image$
end function

function AugmentImage($image, factor$)
  $mapping \leftarrow ModifyPercentiles(image, factor)$
  $image \leftarrow Transform(image, mapping)$
  $image \leftarrow (image - mean(image))/std(image)$ ▷ Mean-variance normalization
  return $image$
end function

Appendix C All results case 1

C.1 Case 1.1

Table C.1: Run 1-5. Training on site A, validating and testing on all sites. Best results were achieved for run 2.

	Run 1				Run 2			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.97859	0.00131	0.97503	0.98057	0.97692	0.00118	0.97390	0.97874
Validation	0.85849	0.09887	0.49348	0.95051	0.88574	0.06153	0.54465	0.94907
A	0.91378	0.04517	0.77593	0.95051	0.90728	0.04986	0.79755	0.94907
B	0.89348	0.06299	0.63323	0.94178	0.89627	0.05295	0.66860	0.93923
C	0.81081	0.10700	0.49348	0.92584	0.87668	0.04613	0.72901	0.93298
D	0.82265	0.17702	0.50789	0.92464	0.82302	0.16013	0.54465	0.93382
Test	0.89151	0.06734	0.65807	0.94461	0.90112	0.03630	0.80073	0.94322
A	0.91504	0.02658	0.88644	0.93898	0.90633	0.03028	0.87341	0.93300
B	0.90925	0.04833	0.76275	0.94461	0.91040	0.03266	0.82139	0.94322
C	0.84391	0.10578	0.65807	0.91616	0.88192	0.04707	0.80073	0.91721
D	0.84600	-	0.84600	0.84600	0.87006	-	0.87006	0.87006

	Run 3				Run 4			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.97672	0.00124	0.97250	0.97835	0.97178	0.00176	0.96700	0.97440
Validation	0.85293	0.09261	0.46300	0.95194	0.86340	0.08482	0.42899	0.94902
A	0.91057	0.05013	0.75298	0.95194	0.90459	0.05596	0.73791	0.94902
B	0.89238	0.06415	0.61388	0.93879	0.89621	0.05224	0.70252	0.93932
C	0.80107	0.08058	0.57904	0.89984	0.82691	0.07470	0.62695	0.90791
D	0.81061	0.19581	0.46300	0.92416	0.79669	0.21080	0.42899	0.92822
Test	0.88862	0.05604	0.73047	0.94763	0.89462	0.04570	0.78542	0.94230
A	0.91528	0.02697	0.89167	0.94467	0.90942	0.02667	0.88821	0.93937
B	0.90693	0.04506	0.77277	0.94763	0.91006	0.04147	0.78558	0.94230
C	0.83372	0.06548	0.73047	0.89925	0.85760	0.04761	0.78542	0.89342
D	0.86345	-	0.86345	0.86345	0.85000	-	0.85000	0.85000

	Run 5			
Set	**Mean**	**SD**	**Min**	**Max**
Training	0.94148	0.00649	0.92540	0.95137
Validation	0.76539	0.17609	0.39427	0.94609
A	0.91205	0.02229	0.86159	0.94609
B	0.88996	0.05067	0.70013	0.93088
C	0.59338	0.12808	0.39427	0.85266
D	0.78494	0.19779	0.44149	0.91101
Test	0.82576	0.14967	0.37822	0.94194
A	0.91808	0.02649	0.88957	0.94194
B	0.90242	0.02553	0.84158	0.93664
C	0.58726	0.12135	0.37822	0.67198
D	0.82137	-	0.82137	0.82137

C.2 Case 1.2

Table C.2: Run 1-5. Training on site C, validating and testing on all sites. Best results were achieved for run 5.

	Run 1				Run 2			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.97001	0.00198	0.96504	0.97353	0.95499	0.00896	0.91337	0.96327
Validation	0.59886	0.08888	0.40867	0.94222	0.62297	0.08977	0.37708	0.93683
A	0.60196	0.06935	0.44508	0.70497	0.60046	0.08522	0.37708	0.71806
B	0.58687	0.07254	0.40867	0.72386	0.62750	0.06601	0.43307	0.74345
C	0.94075	0.00207	0.93929	0.94222	0.93256	0.00604	0.92829	0.93683
D	0.51751	0.08940	0.42544	0.63470	0.66358	0.07519	0.56183	0.73893
Test	0.60541	0.10302	0.41284	0.88120	0.61705	0.11521	0.40091	0.92102
A	0.57638	0.08855	0.41284	0.71037	0.57142	0.08374	0.40091	0.70881
B	0.59589	0.03303	0.56470	0.65964	0.62775	0.04602	0.55968	0.68964
C	-	-	-	-	-	-	-	-
D	0.82260	0.08287	0.76400	0.88120	0.88147	0.05593	0.84192	0.92102

	Run 3				Run 4			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.96071	0.00287	0.95297	0.96610	0.92584	0.03566	0.80072	0.96414
Validation	0.56211	0.09813	0.34265	0.93240	0.59543	0.10836	0.21832	0.93892
A	0.54242	0.09118	0.34265	0.70229	0.58932	0.10468	0.21832	0.74314
B	0.55731	0.06550	0.43470	0.74207	0.57626	0.08391	0.40381	0.73293
C	0.93227	0.00019	0.93214	0.93240	0.91838	0.02905	0.89784	0.93892
D	0.63781	0.08482	0.58088	0.76384	0.69614	0.11024	0.53763	0.77259
Test	0.56597	0.12421	0.38504	0.89986	0.58372	0.10875	0.41709	0.81884
A	0.53093	0.08164	0.38504	0.65314	0.55354	0.08944	0.41709	0.70447
B	0.53636	0.03568	0.48441	0.58147	0.57421	0.07340	0.48370	0.67470
C	-	-	-	-	-	-	-	-
D	0.88253	0.02450	0.86520	0.89986	0.80841	0.01475	0.79799	0.81884

	Run 5			
Set	**Mean**	**SD**	**Min**	**Max**
Training	0.97095	0.00154	0.96668	0.97296
Validation	0.66774	0.07708	0.44111	0.94172
A	0.66484	0.06470	0.44111	0.75671
B	0.66327	0.06145	0.44353	0.74861
C	0.94148	0.00033	0.94125	0.94172
D	0.60790	0.11026	0.45034	0.69623
Test	0.66651	0.06581	0.55388	0.81623
A	0.65286	0.05792	0.55388	0.74564
B	0.66103	0.05941	0.58313	0.75017
C	-	-	-	-
D	0.77174	0.06291	0.72726	0.81623

Appendix D All results case 2

D.1 Case 2.1

Table D.1: Run 1-5. Training on site A, validating and testing on all sites. Best results were achieved for run 1.

	Run 1				Run 2			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.94655	0.00636	0.93035	0.95578	0.94423	0.00609	0.92683	0.95488
Validation	0.92673	0.01950	0.83620	0.95724	0.92423	0.01966	0.83632	0.95311
A	0.93440	0.01532	0.90346	0.95724	0.93004	0.01638	0.89645	0.95311
B	0.92209	0.02212	0.83620	0.94629	0.92094	0.02136	0.83632	0.94392
C	0.93104	0.01115	0.89951	0.94798	0.92809	0.01501	0.87597	0.94669
D	0.90747	0.03829	0.84024	0.93235	0.90369	0.03197	0.85082	0.92823
Test	0.93013	0.01116	0.91175	0.94796	0.92802	0.01221	0.90650	0.94674
A	0.92684	0.00840	0.91789	0.93455	0.92529	0.01715	0.90928	0.94339
B	0.93167	0.01139	0.91175	0.94796	0.93045	0.00958	0.91352	0.94674
C	0.93159	0.01227	0.91807	0.94731	0.92764	0.01524	0.90650	0.94493
D	0.91437	-	0.91437	0.91437	0.90900	-	0.90900	0.90900

	Run 3				Run 4			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.94714	0.00578	0.93262	0.95548	0.93614	0.00774	0.91901	0.94760
Validation	0.92589	0.01861	0.83900	0.95745	0.91397	0.02857	0.77233	0.94564
A	0.93390	0.01629	0.89793	0.95745	0.92452	0.01706	0.89043	0.94564
B	0.92104	0.02035	0.83900	0.94674	0.91303	0.02218	0.82652	0.94158
C	0.93022	0.01203	0.89864	0.95049	0.91473	0.02899	0.80913	0.94454
D	0.90693	0.03247	0.85081	0.93005	0.88757	0.06558	0.77233	0.92741
Test	0.92951	0.01116	0.91110	0.94929	0.91989	0.01077	0.90336	0.93934
A	0.92595	0.01133	0.91522	0.93780	0.91833	0.01291	0.90476	0.93045
B	0.93094	0.01009	0.91110	0.94750	0.92118	0.01196	0.90336	0.93934
C	0.93100	0.01449	0.91571	0.94929	0.91952	0.00874	0.90687	0.92903
D	0.91546	-	0.91546	0.91546	0.91093	-	0.91093	0.91093

	Run 5			
Set	**Mean**	**SD**	**Min**	**Max**
Training	0.93623	0.00734	0.91209	0.95135
Validation	0.91027	0.03696	0.65970	0.94930
A	0.92215	0.02097	0.87241	0.94930
B	0.91199	0.02822	0.79609	0.93545
C	0.91222	0.02506	0.81065	0.93820
D	0.85296	0.10965	0.65970	0.92637
Test	0.91713	0.01812	0.86955	0.94354
A	0.91950	0.01659	0.90123	0.93362
B	0.91801	0.02074	0.86955	0.94354
C	0.91777	0.01436	0.89501	0.93233
D	0.89637	-	0.89637	0.89637

D.2 Case 2.2

Table D.2: Run 1-5. Training on site C, validating and testing on all sites. Best results were achieved for run 4.

	Run 1				Run 2			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.94740	0.00611	0.93337	0.95793	0.94763	0.00544	0.93525	0.95667
Validation	0.91798	0.02348	0.80989	0.95001	0.91819	0.02425	0.80030	0.95145
A	0.91926	0.01904	0.84764	0.95001	0.92013	0.02025	0.83053	0.95145
B	0.91752	0.02612	0.80989	0.94312	0.91627	0.02698	0.80030	0.94504
C	0.93929	0.00860	0.93321	0.94537	0.94070	0.00903	0.93431	0.94709
D	0.89830	0.03613	0.84812	0.93329	0.90571	0.03632	0.85569	0.93773
Test	0.91894	0.01412	0.88094	0.94152	0.91734	0.01526	0.87782	0.94782
A	0.91757	0.01424	0.88094	0.93697	0.91504	0.01521	0.87782	0.93792
B	0.91995	0.01618	0.90570	0.94152	0.91993	0.01760	0.89951	0.94782
C	-	-	-	-	-	-	-	-
D	0.92482	0.01191	0.91640	0.93324	0.92459	0.01079	0.91696	0.93222

	Run 3				Run 4			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.94037	0.00640	0.92590	0.95448	0.95191	0.00494	0.94069	0.96132
Validation	0.91207	0.02582	0.80268	0.94809	0.91904	0.02428	0.80562	0.94888
A	0.91706	0.01798	0.86102	0.94809	0.92050	0.02169	0.81321	0.94888
B	0.90759	0.02927	0.80268	0.93866	0.91719	0.02692	0.80562	0.94690
C	0.93601	0.00917	0.92953	0.94249	0.94064	0.00706	0.93565	0.94563
D	0.89245	0.04769	0.82319	0.93024	0.91143	0.02751	0.87328	0.93859
Test	0.91687	0.01580	0.87576	0.93509	0.91985	0.01611	0.87025	0.94165
A	0.91898	0.01495	0.87576	0.93509	0.91675	0.01843	0.87025	0.94165
B	0.90955	0.01823	0.88717	0.93253	0.92501	0.01177	0.91344	0.94013
C	-	-	-	-	-	-	-	-
D	0.92508	0.01151	0.91694	0.93321	0.92452	0.00984	0.91756	0.93148

	Run 5			
Set	**Mean**	**SD**	**Min**	**Max**
Training	0.94185	0.00624	0.92943	0.95385
Validation	0.91325	0.02794	0.79469	0.94458
A	0.91597	0.02004	0.84341	0.94457
B	0.91197	0.03014	0.80576	0.94458
C	0.93709	0.00545	0.93324	0.94095
D	0.88526	0.06214	0.79469	0.93195
Test	0.91334	0.01442	0.89078	0.94047
A	0.91132	0.01378	0.89078	0.93524
B	0.91613	0.01726	0.89670	0.94047
C	-	-	-	-
D	0.91814	0.01568	0.90706	0.92923

Appendix E All results case 3

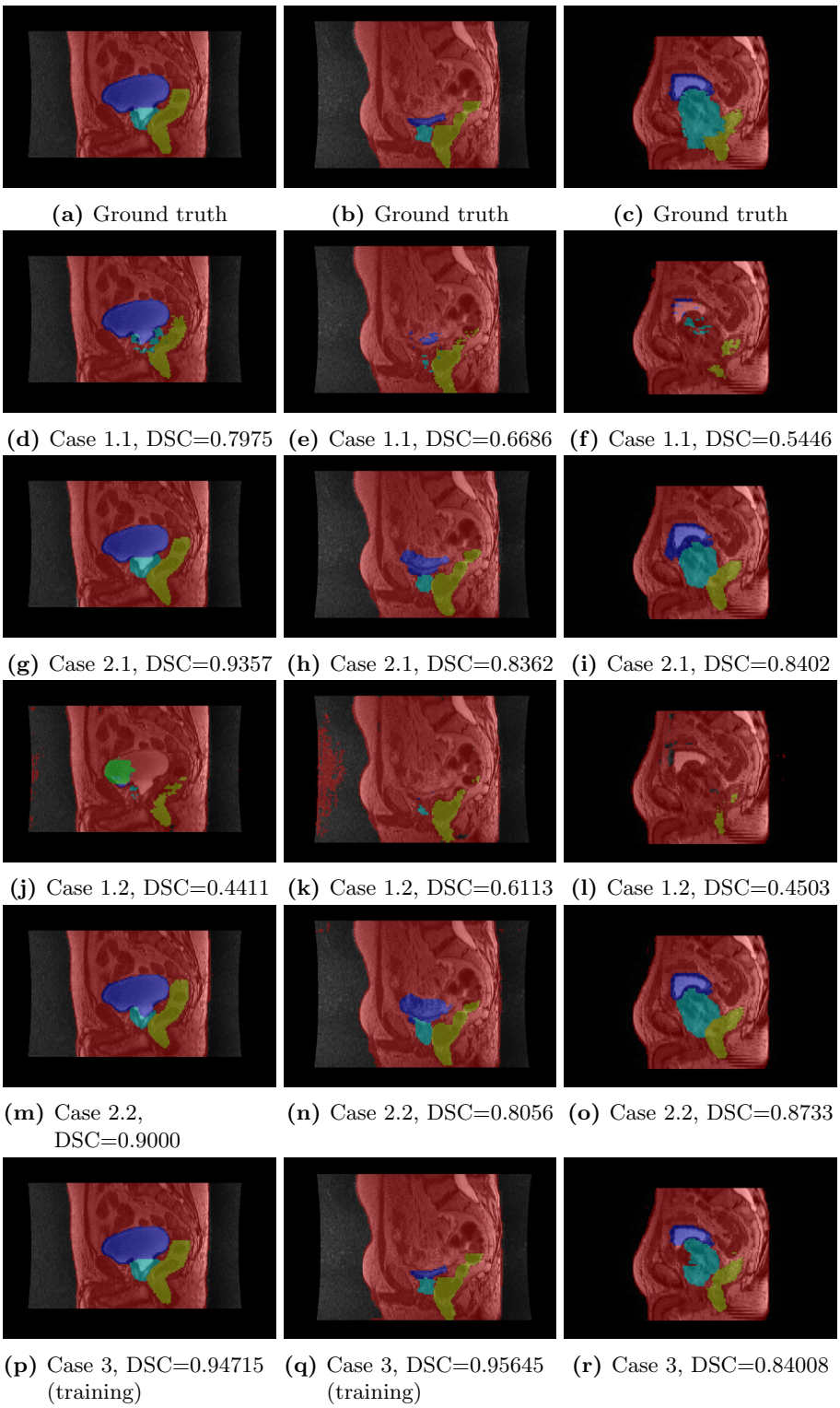
Table E.1: Run 1-5. Training, validating and testing on all sites. Best results were achieved for run 4.

	Run 1				Run 2			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.96885	0.00215	0.96168	0.97368	0.95753	0.00408	0.94243	0.96543
A	0.96939	0.00180	0.96420	0.97233	0.95835	0.00385	0.94669	0.96473
B	0.96899	0.00266	0.96168	0.97368	0.95633	0.00477	0.94243	0.96293
C	0.96787	0.00170	0.96470	0.97095	0.95763	0.00344	0.94999	0.96543
D	0.96818	0.00232	0.96604	0.97065	0.95730	0.00338	0.95385	0.96059
Validation	0.92909	0.02354	0.81834	0.95233	0.92617	0.02664	0.80352	0.94721
A	0.93625	0.00930	0.92219	0.95233	0.93790	0.00899	0.92149	0.94721
B	0.93068	0.01940	0.87668	0.95012	0.92673	0.01752	0.88075	0.94300
C	0.93077	0.01291	0.90067	0.94637	0.92594	0.01981	0.87401	0.94707
D	0.87082	0.07422	0.81834	0.92330	0.86005	0.07995	0.80352	0.91658
Test	0.93023	0.01291	0.91098	0.95743	0.93080	0.01459	0.91080	0.95707
A	0.92732	0.01193	0.91098	0.93621	0.92987	0.01119	0.91551	0.94043
B	0.92784	0.01136	0.91443	0.94122	0.92735	0.01287	0.91246	0.94222
C	0.93917	0.01608	0.91858	0.95743	0.93995	0.02014	0.91080	0.95707
D	0.92042	-	0.92042	0.92042	0.91858	-	0.91858	0.91858

	Run 3				Run 4			
Set	**Mean**	**SD**	**Min**	**Max**	**Mean**	**SD**	**Min**	**Max**
Training	0.95640	0.00410	0.94312	0.96502	0.96500	0.00269	0.95645	0.97072
A	0.95646	0.00448	0.94312	0.96501	0.96587	0.00252	0.95850	0.96956
B	0.95587	0.00443	0.94776	0.96502	0.96454	0.00291	0.95645	0.97072
C	0.95706	0.00311	0.94944	0.96270	0.96436	0.00238	0.95828	0.96956
D	0.95531	0.00319	0.95276	0.95888	0.96322	0.00267	0.96074	0.96604
Validation	0.92872	0.01962	0.86278	0.94990	0.93002	0.02063	0.84008	0.94937
A	0.93545	0.00861	0.91985	0.94990	0.93670	0.00815	0.92365	0.94888
B	0.92778	0.02274	0.86738	0.94627	0.93326	0.01716	0.88883	0.94813
C	0.93012	0.01498	0.89324	0.94386	0.92928	0.01500	0.89432	0.94937
D	0.88851	0.03639	0.86278	0.91424	0.87995	0.05638	0.84008	0.91982
Test	0.92934	0.01387	0.90871	0.95716	0.93124	0.01450	0.91262	0.95473
A	0.92857	0.01043	0.91872	0.94286	0.93024	0.01413	0.91262	0.94289
B	0.92746	0.01069	0.90887	0.93751	0.92744	0.01350	0.91314	0.94349
C	0.93810	0.01861	0.91291	0.95716	0.94244	0.01309	0.92396	0.95473
D	0.90871	-	0.90871	0.90871	0.91318	-	0.91318	0.91318

	Run 5			
Set	**Mean**	**SD**	**Min**	**Max**
Training	0.95850	0.00360	0.94881	0.96495
A	0.95863	0.00346	0.94881	0.96461
B	0.95775	0.00420	0.94981	0.96495
C	0.95912	0.00307	0.95249	0.96337
D	0.95903	0.00362	0.95542	0.96265
Validation	0.92763	0.03196	0.76537	0.94869
A	0.93736	0.00722	0.92644	0.94869
B	0.93097	0.01957	0.87929	0.94667
C	0.93000	0.01841	0.88443	0.94726
D	0.84150	0.10766	0.76537	0.91763
Test	0.93103	0.01572	0.91238	0.95645
A	0.92980	0.01289	0.91420	0.94231
B	0.92890	0.01598	0.91238	0.94866
C	0.93982	0.01864	0.91319	0.95645
D	0.91364	-	0.91364	0.91364

Appendix F Result: images in the sagittal plane



(a) Ground truth **(b)** Ground truth **(c)** Ground truth

(d) Case 1.1, DSC=0.7975 **(e)** Case 1.1, DSC=0.6686 **(f)** Case 1.1, DSC=0.5446

(g) Case 2.1, DSC=0.9357 **(h)** Case 2.1, DSC=0.8362 **(i)** Case 2.1, DSC=0.8402

(j) Case 1.2, DSC=0.4411 **(k)** Case 1.2, DSC=0.6113 **(l)** Case 1.2, DSC=0.4503

(m) Case 2.2, DSC=0.9000 **(n)** Case 2.2, DSC=0.8056 **(o)** Case 2.2, DSC=0.8733

(p) Case 3, DSC=0.94715 (training) **(q)** Case 3, DSC=0.95645 (training) **(r)** Case 3, DSC=0.84008

Figure F.1: Comparison of segmentations for three different patients from site B, C, and D, respectively. The images are from the validation set if not otherwise stated. Original MR images courtesy of Gentle Radiotherapy.

Appendix G Organ comparison case 1-3

Table G.1: Mean, standard deviation, minimum, and maximum DSC for the different organs across the validation sets for case 1 - 3.

	Case 1.1	**Case 1.2**	**Case 2.1**	**Case 2.2**	**Case 3**
Organ			**Mean**		
Body	0.98934	0.94290	0.99087	0.98727	0.99324
Femoral heads	0.94390	0.87285	0.94990	0.94939	0.95434
Bladder	0.90514	0.07838	0.94067	0.93008	0.95173
Rectum	0.79368	0.77633	0.87401	0.85958	0.87425
Prostate	0.79662	0.66823	0.87822	0.86886	0.87187
			SD		
Body	0.00604	0.05177	0.00259	0.00595	0.00155
Femoral heads	0.01498	0.08044	0.00771	0.00923	0.00548
Bladder	0.10735	0.14138	0.05705	0.07583	0.02991
Rectum	0.12344	0.14317	0.04966	0.06092	0.04543
Prostate	0.14259	0.21051	0.03577	0.04003	0.06197
			Min		
Body	0.95593	0.76290	0.98121	0.96620	0.98735
Femoral heads	0.83758	0.69172	0.92146	0.90469	0.93903
Bladder	0.31950	0.00132	0.49706	0.39506	0.84882
Rectum	0.31357	0.27534	0.69925	0.64308	0.71834
Prostate	0.10740	0.00060	0.77878	0.73801	0.58691
			Max		
Body	0.99445	0.99458	0.99374	0.99508	0.99480
Femoral heads	0.95965	0.96223	0.96312	0.96265	0.96229
Bladder	0.97225	0.96940	0.97753	0.97387	0.97672
Rectum	0.91836	0.92237	0.93663	0.93335	0.92371
Prostate	0.93063	0.90065	0.94614	0.92879	0.93129

Appendix H Side projects

The following experiments have developed along side the three main cases for this thesis. The motivation behind these projects was either a potential improvement in performance if applied to the main cases or more investigation of augmentation effects. It should be noted that these side projects were not tested to the same extent as Case 1 - 3.

H.1 Extreme augmentation on small data set

A training set consisting of four image volumes, one from each site, were randomly selected. The rest of the data set was split into validation, 90 image volumes, and test, 57 volumes. Higher probabilities for augmentation than in the main cases were used, compensating for the small data set.

There was a difference in network parameters compared to Case 1 - 3 that affected network performance. This was due to that only the computer with NVIDIA GeForce GTX 1080 was used, resulting in memory restrictions and that only a batch-size of one could be used. The network was trained 20 000 iterations.

The following augmentation probabilities were used: instead of 50%, the probability for applying histogram augmentation, bias field, and adding noise was 80%, and 40% for smoothing. The geometric augmentation layers were still set to 50% (except for elastic deformation, which could not be used due to software restrictions). See Appendix A for further parameter specifications.

H.1.1 Results and discussion

The mean DSC on the validation set was 0.796 without augmentation and 0.880 with augmentation. The mean values cannot be said to differ between the sites in either of the experiments, but site D had a considerably lower mean value. The standard deviations were lower for all sets and sites, except for validation for site D, when augmentation was used compared to no augmentation. For details, see Table H.1. Minimum values increased with $0.066 - 0.335$ when adding augmentation except for site D where it decreased with 0.057.

Table H.1: DSC for training, validation, and test sets for the side project with four image volumes in training together with the validation result for each site. The mean, minimum, and maximum values, as well as standard deviation for the DSC across the data sets are presented.

	No augmentation					**Augmentation**				
Set	**N**	**Mean**	**SD**	**Min**	**Max**	**N**	**Mean**	**SD**	**Min**	**Max**
Training	4	0.97135	0.00429	0.96583	0.97620	4	0.94477	0.00268	0.94212	0.94850
Validation	90	0.79634	0.12541	0.41158	0.91940	90	0.88050	0.05860	0.51610	0.93318
A	35	0.79076	0.11861	0.51861	0.91117	35	0.88674	0.03912	0.70548	0.93318
B	26	0.76215	0.16553	0.41158	0.90251	26	0.87837	0.04759	0.74664	0.92431
C	27	0.84489	0.05935	0.69138	0.91940	27	0.89065	0.04588	0.75696	0.92828
D	2	0.68315	0.15551	0.57318	0.79311	2	0.66219	0.20660	0.51610	0.80828
Test	57	0.77505	0.13523	0.44608	0.92130	57	0.87759	0.05804	0.54485	0.92717

It is interesting to note that training on only four multi-site image stacks and including augmentation yields similar results as training on 40 image stacks from site A without augmentations. It is also not too far from the mean validation score when comparing to case 3 (training on 100 images), 0.880 vs 0.930, but the minimum values and standard deviations are not similar. It is worth to emphasize, once again, that this side project is not directly comparable to case 3 since this project was only trained with a batch-size of one and no elastic deformation. These are two parameters that affect the performance on outliers and thus also the standard deviation.

H.2 Augmentation in inference

Due to rather heavy augmentation during training, and none during inference, experiments with adding augmentation in inference were also conducted.

As the augmentation sometimes could produce a rather distorted image, the inference was performed 11 times, i.e., 11 segmentations for each patient was obtained. The 11 segmentations were combined into a final segmentation by label voting. This means that for each pixel, each segmentation "votes" on which label the pixel should belong to, and the label with most votes wins. For pixels with equal amount of votes, we set the default value to one which corresponds to the body segment.

When applying augmentation in inference, only the histogram augmentation, bias field, and Gaussian noise layers were applied with a probability of 40% for each layer. This experiment was performed on the best models from case 2.1 and 2.2. See Appendix A for further parameter specifications.

H.2.1 Results and discussion

When adding augmentation layers in the inference and then inferring multiple times for the best model in case 2.1 and 2.2, the results were very similar to the ones presented earlier. The DSC for case 2.1 was 0.927 and 0.919 for case 2.2. The results can be found in Table H.2.

Table H.2: DSC for training, validation, and test sets for case 2.1 and 2.2 when adding augmentation in the inference step. The mean, minimum, and maximum values as well as standard deviation for the DSC across the data sets are presented.

	Case 2.1					**Case 2.2**				
Set	**N**	**Mean**	**SD**	**Min**	**Max**	**N**	**Mean**	**SD**	**Min**	**Max**
Training	40	0.94671	0.00635	0.93026	0.95639	40	0.95190	0.00494	0.94031	0.96106
Validation	90	0.92701	0.01947	0.83600	0.95741	90	0.91915	0.02434	0.80626	0.94837
A	13	0.93456	0.01523	0.90346	0.95741	43	0.92033	0.02186	0.81221	0.94837
B	37	0.92247	0.02214	0.83600	0.94696	41	0.91762	0.02693	0.80626	0.94767
C	35	0.93123	0.01128	0.89906	0.94781	2	0.94062	0.00706	0.93563	0.94562
D	5	0.90789	0.03796	0.84124	0.93244	4	0.91144	0.02746	0.87342	0.93870
Test	21	0.93030	0.01139	0.91194	0.94826	21	0.92016	0.01575	0.87215	0.94057

The reason for the indifferent result could be that the method has not been tested enough and that the best set-up has not been found. It could also indicate that the networks from case 2.1 and 2.2, which have been exposed to augmented images during training are robust to these types of changes, and therefore produce similar results for an augmented image as for the original one.

H.3 Separate networks for different organs

For all previously mentioned experiments, the network was given six labels to predict (background, body, femoral heads, bladder, rectum, and prostate) through a single model. An idea of how to improve the segmentation results was to split the segmentation task between two different networks; one for the air and body, and one for the remaining labels.

The networks were trained separately with different configuration files. The data split was the same as the one for case 2.2. The OAR network's prediction was then overlaid on the body network's prediction. These networks were trained both with, and without augmentation.

For this side project, a batch-size of one was used. Other parameters are specified in Appendix A.

H.3.1 Result and discussion

Using separate networks for the body and remaining organs did not improve DSC when compared to case 1.2 and case 2.2, respectively. The result on the validation set without any augmentation was 0.495 with a standard deviation of 0.084, which is lower than for case 1.2. Adding augmentation resulted in a DSC of 0.848 on the validation set with 0.10 as standard deviation, which is lower than the original inference results. The detailed results are found in Table H.3.

The reason why this did not improve the results, could be due to using a batch-size of one and no elastic deformation. It is also reasonable that the networks need more tuning, as the networks created from case 2 that were optimized for six labels served as a starting point.

Table H.3: DSC for training, validation, and test sets when using two separate networks, both with and without augmentation. The mean, minimum, and maximum values, as well as standard deviation for the DSC across the data sets are presented.

	No augmentation					**Augmentation**				
Set	**N**	**Mean**	**SD**	**Min**	**Max**	**N**	**Mean**	**SD**	**Min**	**Max**
Training	40	0.94321	0.00871	0.91873	0.95553	40	0.93434	0.00762	0.91623	0.94560
Validation	90	0.49503	0.08354	0.37721	0.93578	90	0.84789	0.10050	0.52721	0.93335
A	43	0.48012	0.05740	0.38196	0.61917	43	0.87600	0.06475	0.68451	0.93257
B	41	0.49390	0.04424	0.37721	0.58033	41	0.81785	0.11444	0.56120	0.92840
C	2	0.93320	0.00365	0.93062	0.93578	2	0.93235	0.00141	0.93135	0.93335
D	4	0.44794	0.02546	0.41433	0.47598	4	0.81153	0.19019	0.52721	0.92614
Test	21	0.48278	0.09039	0.36163	0.80863	21	0.84923	0.08255	0.63124	0.93590

Bibliography

[1] Emilia Persson et al. "MR-OPERA: A Multicenter/Multivendor Validation of Magnetic Resonance Imaging–Only Prostate Treatment Planning Using Synthetic Computed Tomography Images". In: *International Journal of Radiation Oncology, Biology, Physics* 99.3 (2017), pp. 692–700. ISSN: 0360-3016. DOI: https://doi.org/10.1016/j.ijrobp.2017.06.006.

[2] *Analytic Imaging Diagnostics Arena, AIDA*. URL: https://medtech4health.se/aida/.

[3] *Cancer i siffror 2018 – Populärvetenskapliga fakta om cancer*. Popular Science Publication. Cancerfonden and Socialstyrelsen, 2018. URL: https://www.cancerfonden.se/cancer-i-siffror.

[4] *Nationella riktlinjer för bröst-, prostata-, tjocktarms- och ändtarmscancervård*. Treatment Guidelines. Socialstyrelsen, 2014. URL: https://www.socialstyrelsen.se/regler-och-riktlinjer/nationella-riktlinjer/slutliga-riktlinjer/brost-prostata-tjocktarms-och-andtarmscancervard/.

[5] *Prostatacancer - Nationell kvalitetsrapport för 2016*. Quality report. Nationella Prostatacancerregistret, 2017. URL: http://npcr.se/rapporter/nationella-arsrapporter/.

[6] F. Milletari, N. Navab, and S. Ahmadi. "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation". In: *2016 Fourth International Conference on 3D Vision (3DV)*. Oct. 2016, pp. 565–571. DOI: 10.1109/3DV.2016.79.

[7] Adegun Adekanmi Adeyinka and Serestina Viriri. "Skin Lesion Images Segmentation: A Survey of the State-of-the-Art". In: *Mining Intelligence and Knowledge Exploration*. Ed. by Adrian Groza and Rajendra Prasath. Cham: Springer International Publishing, 2018, pp. 321–330. ISBN: 978-3-030-05918-7.

[8] *Gentle Radiotherapy*. URL: http://gentleradiotherapy.se.

[9] Rifat Atun Trishan Panch Peter Szolovits. "Artificial intelligence, machine learning and health systems". In: *J Glob Health* 8 (2) (2018). DOI: 10.7189/jogh.08.020303.

[10] A.P. Vassilopoulos and E.F. Georgopoulos. "5 - Novel computational methods for fatig life modeling of composite materials". In: *Fatigue Life Prediction of Composites and Composite Structures*. Ed. by Anastasios P. Vassilopoulos. Woodhead Publishing Series in Composites Science and Engineering. Woodhead Publishing, 2010, pp. 139–173. ISBN: 978-1-84569-525-5. DOI: https://doi.org/10.1533/9781845699796.1.139.

[11] Oludare Isaac Abiodun et al. "State-of-the-art in artificial neural network applications: A survey". In: *Heliyon* 4 (11) (2018). DOI: https://doi.org/10.1016/j.heliyon.2018.e00938.

[12] Rikiya Yamashita et al. "Convolutional neural networks: an overview and application in radiology". In: *Insights Imaging* 9:611 (2018). ISSN: 1869-4101. DOI: https://doi.org/10.1007/s13244-018-0639-9.

[13] David J. Livingstone. *Artificial Neural Networks: Methods and Application*. ISBN:978-1-58829-718-1, doi: https://doi.org/10.1007/978-1-60327-101-1. Humana Press, 2009.

[14] S. Samarasinghe S. Shanmuganathan. *Artificial Neural Network Modelling*. ISBN:978-3-319-28493-4, doi: https://doi.org/10.1007/978-3-319-28495-8_1. Springer, Cham, 2016.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[16] T. Yamashita, H. Furukawa, and H. Fujiyoshi. "Multiple Skip Connections of Dilated Convolution Network for Semantic Segmentation". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. Oct. 2018, pp. 1593–1597. DOI: 10.1109/ICIP.2018.8451033.

[17] Vincent Dumoulin and Francesco Visin. "A guide to convolution arithmetic for deep learning". In: *arXiv e-prints*, arXiv:1603.07285 (2016). URL: https://arxiv.org/abs/1603.07285.

[18] Léon Bouttou. "Large-Scale Machine Learning with Stochastic Gradient Descent". In: *Proceedings of COMPSTAT'2010* (2010). DOI: https://doi.org/10.1007/978-3-7908-2604-3_16.

[19] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *CoRR* (2016). URL: http://arxiv.org/abs/1609.04747.

[20] Ashia C Wilson et al. "The Marginal Value of Adaptive Gradient Methods in Machine Learning". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4148–4158. URL: http://papers.nips.cc/paper/7003-the-marginal-value-of-adaptive-gradient-methods-in-machine-learning.pdf.

[21] Z. Zhang. "Improved Adam Optimizer for Deep Neural Networks". In: *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. 2018. DOI: 10.1109/IWQoS.2018.8624183.

[22] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015. URL: http://arxiv.org/abs/1412.6980.

[23] N. Tajbakhsh et al. "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?" In: *IEEE Transactions on Medical Imaging* 35.5 (May 2016), pp. 1299–1312. ISSN: 0278-0062. DOI: 10.1109/TMI.2016.2535302.

[24] Bing Xu et al. "Empirical Evaluation of Rectified Activations in Convolutional Network". In: *CoRR* (2015). URL: http://arxiv.org/abs/1505.00853.

[25] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *CoRR* abs/1502.03167 (2015). URL: http://arxiv.org/abs/1502.03167.

[26] Abdel Aziz Taha and Allan Hanbury. "Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool". In: *BMC Medical Imaging* 15.1 (Aug. 2015). ISSN: 1471-2342. DOI: 10.1186/s12880-015-0068-x.

[27] *Hausdorff Distance*. URL: https://en.wikipedia.org/wiki/Hausdorff_distance.

[28] W. R. Crum, O. Camara, and D. L. G. Hill. "Generalized Overlap Measures for Evaluation and Validation in Medical Image Analysis". In: *IEEE Transactions on Medical Imaging* 25.11 (Nov. 2006), pp. 1451–1461. ISSN: 0278-0062. DOI: 10.1109/TMI.2006.880587.

[29] Perry Sprawls. *Magnetic Resonance Imaging: Principles, Methods, and Techniques*. Available online at http://www.sprawls.org/mripmt/. Medical Physics Publishing, 2000.

[30] Hákon Gudbjartsson and Samuel Patz. "The rician distribution of noisy mri data". In: *Magnetic Resonance in Medicine* 34.6 (1995), pp. 910–914. DOI: 10.1002/mrm.1910340618.

[31] *Normal Distribution*. 2019. URL: https://en.wikipedia.org/wiki/Normal_distribution (visited on 05/10/2019).

[32] *Rayleigh Distribution*. 2019. URL: https://en.wikipedia.org/wiki/Rayleigh_distribution (visited on 05/10/2019).

[33] *Rice Distribution*. 2019. URL: https://en.wikipedia.org/wiki/Rice_distribution (visited on 05/10/2019).

[34] Hou Zujun. "A Review on MR Image Intensity Inhomogeneity Correction." In: *International Journal of Biomedical Imaging* (2006). ISSN: 1687-4188. DOI: 10.1155/IJBI/2006/49515.

[35] Peter Tuominen. *Undersökning med magnetkamera*. 2018. URL: https://www.1177.se/Skane/behandling--hjalpmedel/undersokningar-och-provtagning/bildundersokningar-och-rontgen/magnetkameraundersokning/ (visited on 05/10/2019).

[36] Jiayu Chen et al. "Exploration of scanning effects in multi-site structural MRI studies". In: *Journal of Neuroscience Methods* 230 (2014), pp. 37–50. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2014.04.023.

[37] Tong Zhu et al. "Quantification of accuracy and precision of multi-center DTI measurements: A diffusion phantom and human brain study". In: *NeuroImage* 56.3 (2011), pp. 1398–1411. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2011.02.010.

[38] Harrison Nguyen et al. "Correcting differences in multi-site neuroimaging data using Generative Adversarial Networks". In: 2018. URL: http://arxiv.org/abs/1803.09375.

[39] Anthony Doemer et al. "Evaluating organ delineation, dose calculation and daily localization in an open-MRI simulation workflow for prostate cancer patients". In: *Radiation Oncology* 10:37 (2015). DOI: 10.1186/s13014-014-0309-0.

[40] Neil G Burnet et al. "Defining the tumour and target volumes for radiotherapy." In: *Cancer Imaging: The Official Publication Of The International Cancer Imaging Society* 4.2 (2004), pp. 153–161. ISSN: 1470-7330. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1434601/.

[41] J. Korhonen et al. "Clinical Experiences of Treating Prostate Cancer Patients With Magnetic Resonance Imaging–Only Based Radiation Therapy Treatment Planning Workflow". In: *International Journal of Radiation Oncology, Biology, Physics* 96.2, Supplement (2016). Proceedings of the American Society for Radiation Oncology, S225. ISSN: 0360-3016. DOI: https://doi.org/10.1016/j.ijrobp.2016.06.558.

[42] Marisol De Brabandere et al. "Prostate post-implant dosimetry: Interobserver variability in seed localisation, contouring and fusion". In: *Radiotherapy and Oncology* 104.2 (2012), pp. 192–198. ISSN: 0167-8140. DOI: https://doi.org/10.1016/j.radonc.2012.06.014.

[43] Carl Salembier et al. "ESTRO ACROP consensus guideline on CT- and MRI-based target volume delineation for primary radiation therapy of localized prostate cancer". In: *Radiotherapy and Oncology* 127.1 (2018), pp. 49–61. ISSN: 0167-8140. DOI: https://doi.org/10.1016/j.radonc.2018.01.014.

[44] Sara Thörnqvist et al. “Propagation of target and organ at risk contours in radiotherapy of prostate cancer using deformable image registration.” In: *Acta Oncologica* 49.7 (2010), pp. 1023–1032. ISSN: 0284186X.

[45] Dominique P. Huyskens et al. “A qualitative and a quantitative analysis of an auto-segmentation module for prostate cancer”. In: *Radiotherapy and Oncology* 90.3 (2009), pp. 337–345. ISSN: 0167-8140. DOI: https://doi.org/10.1016/j.radonc.2008.08.007.

[46] Carl Siversson et al. “Technical Note: MRI only prostate radiotherapy planning using the statistical decomposition algorithm”. In: *Medical Physics* 42.10 (2015), pp. 6090–6097. DOI: 10.1118/1.4931417.

[47] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[48] Bradley C. Lowekamp et al. “The Design of SimpleITK.” In: *Frontiers in Neuroinformatics* 7 (2013). ISSN: 16625196.

[49] Ziv Yaniv et al. “SimpleITK Image-Analysis Notebooks: a Collaborative Environment for Education and Reproducible Research.” In: *Journal of Digital Imaging* 31.3 (2018), pp. 290–303. ISSN: 0897-1889.

[50] Paul A. Yushkevich et al. “User-Guided 3D Active Contour Segmentation of Anatomical Structures: Significantly Improved Efficiency and Reliability”. In: *Neuroimage* 31.3 (2006), pp. 1116–1128.

[51] Eli Gibson et al. “NiftyNet: a deep-learning platform for medical imaging”. In: vol. 158. 2018, pp. 113–122.

[52] Wenqi Li et al. “On the Compactness, Efficiency, and Representation of 3D Convolutional Networks: Brain Parcellation as a Pretext Task”. In: *CoRR* (2017). arXiv: 1707.01992. URL: http://arxiv.org/abs/1707.01992.

[53] Guodong Zeng and Guoyan Zheng. “Holistic Decomposition Convolution for Effective Semantic Segmentation of 3D MR Images”. In: *CoRR* (2018). URL: http://arxiv.org/abs/1812.09834.

[54] Ben A. Duffy et al. “Retrospective correction of motion artifact affected structural MRI images using deep learning of simulated motion”. In: 2018.

[55] SimpleITK. *BSplineTransformInitializerFilter, Class Reference*. URL: https://itk.org/SimpleITKDoxygen/html/classitk_1_1simple_1_1BSplineTransformInitializerFilter.html.

[56] NiftyNet. *Histogram Standardisation Module*. URL: https://niftynet.readthedocs.io/en/dev/niftynet.utilities.histogram_standardisation.html.

[57] M. Frid-Adar et al. “Synthetic data augmentation using GAN for improved liver lesion classification”. In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. Apr. 2018, pp. 289–293. DOI: 10.1109/ISBI.2018.8363576.

[58] Y. Chi et al. “Controlled Synthesis of Dermoscopic Images via a New Color Labeled Generative Style Transfer Network to Enhance Melanoma Segmentation”. In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. July 2018, pp. 2591–2594. DOI: 10.1109/EMBC.2018.8512842.

[59] Luke Taylor and Geoff Nitschke. “Improving Deep Learning using Generic Data Augmentation”. In: *CoRR* abs/1708.06020 (2017). arXiv: 1708.06020. URL: http://arxiv.org/abs/1708.06020.

[60] *PROMISE12*. URL: https://promise12.grand-challenge.org/.

[61] Lucas Fidon et al. "Generalised Wasserstein Dice Score for Imbalanced Multi-class Segmentation using Holistic Convolutional Networks". In: *CoRR* (2017). URL: http://arxiv.org/abs/1707.00478.