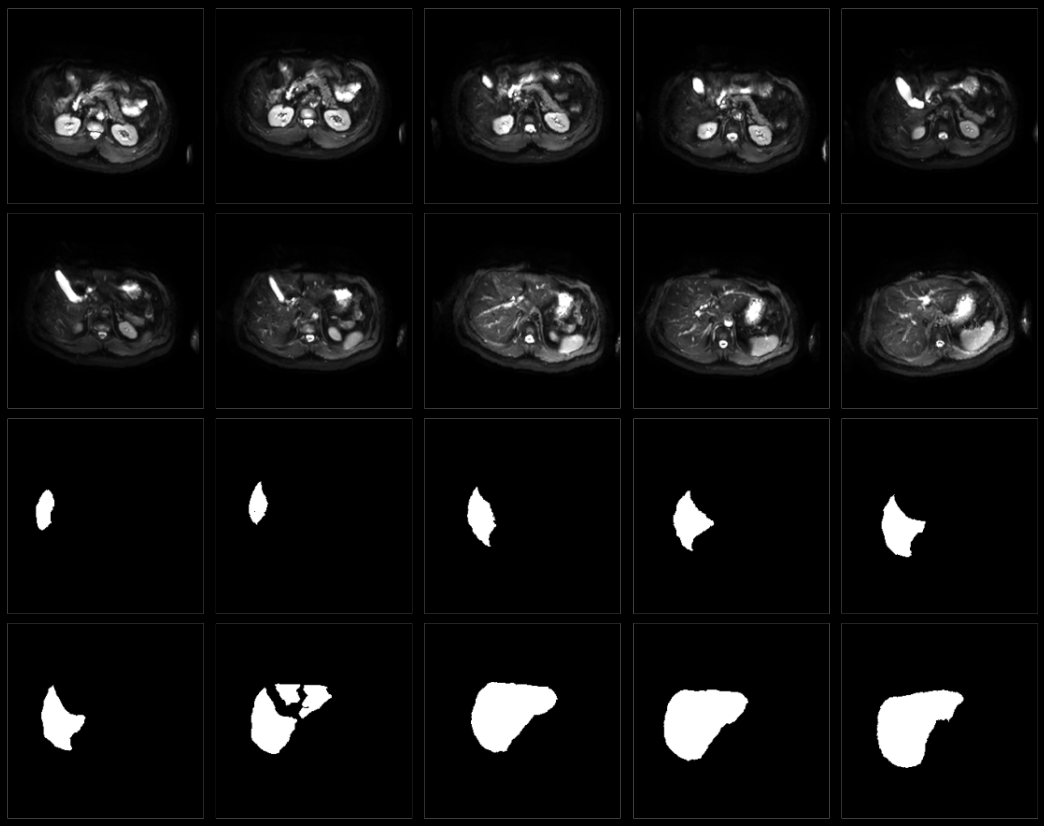
Automatic Liver Segmentation of qMRI images: A Deep Learning Approach



Dilara Tank

Layout: typeset by the author using LaTeX.
Cover illustration: qMRI liver dataset (made by Dilara Tank, provided by AUMC)

Automatic Liver Segmentation of qMRI images: A Deep Learning Approach

Dilara Tank
12170062

Bachelor thesis
Credits: 18 EC

Bachelor *Kunstmatige Intelligentie*

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
Oliver J. Gurney-Champion

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 907
1098 XG Amsterdam

June, 2021

Abstract

Quantitative MRI is a great tool for imaging fatty livers. In qMRI, the intensity of each pixel in the image corresponds to a measurement of a real physical property. In researching the assessment of disease severity along the NAFLD spectrum, this physical property is diffusion, as diffusion significantly correlates with disease activity. However, to obtain quantitative features, such as the diffusion, from medical images, ROIs indicating the tissue of interest (e.g. liver in fatty liver disease) need to be drawn. Currently, this is mainly done by manual analysis, which is time-consuming, labor-intensive, and prone to error. A possible approach for automated contouring of ROIs is using deep learning. For biomedical image segmentation, the U-Net architecture and its variations are widely used. While automatic liver segmentation has been proven to work well on CT and conventional MRI data, only a few studies can be found of automatic segmentation on qMRI data. We hypothesized that segmentation on qMRI data can be automated with a U-Net and SegNet architecture. To investigate this, we trained a U-Net, SegNet, and some architecture variations on data of 37 patients with NAFLD and 15 healthy volunteers (52 in total). The best performing model was the standard U-Net architecture, which received a mean Dice score of 0.91, with which we showed that automatic segmentation on qMRI data can indeed be automated. We propose that training a multi-channel U-Net with different b-values and augmentations could even improve this score.

Keywords: Auto segmentation; Medical Imaging; Deep Learning; U-Net; SegNet; CNN

Contents

CHAPTER 1

Introduction

Quantitative magnetic resonance imaging (qMRI) is a great tool for imaging fatty livers. Fatty liver disease is associated with obesity, excessive alcohol drinking, and diabetes. A liver is considered fatty if more than 5% of it is fat (Ahmed, 2015). There are multiple medical imaging approaches for imaging the liver, each with its strengths and weaknesses, including computed tomography (CT) and MRI. They are widely used for diagnosis, classification of malignancies, treatment stratification, and treatment monitoring (Talbar et al., 2020). Quantitative MRI, which is a sub-form of MRI, obtains maps of quantitative features (physical or chemical variables) that can be measured in physical units, in addition to performing the conventional "qualitative" visual inspection of the images (Pierpaoli, 2010). In diffusion weighted imaging (DWI), the physical variable that is measured is the diffusion of water molecules. It was found that tissue characteristics that are altered in chronic liver diseases, such as nonalcoholic liver fatty disease (NALFD), have an effect on the diffusion in the liver (Murphy et al., 2015; Shenoy-Bhangle, Baliyan, Kordbacheh, Guimaraes, & Kambadakone, 2017). This makes DWI a great tool for distinguishing a healthy liver from a fatty liver without invasive procedures.

However, to obtain quantitative features from medical images, regions of interest (ROIs) indicating the tissue of interest (e.g. liver in fatty liver disease) need to be drawn. In DWI, that means that a segmentation of the tissue of interest has to be made to obtain the diffusion in that area. Currently, this is mainly done by manual analysis, which is time-consuming, labor-intensive, and prone to error (Wang et al., 2019). Automatic contouring of ROIs could be of great help to ensure contouring quality and reduce manual workload and time required for treatment planning (Boldrini, Bibault, Masciocchi, Shen, & Bittner, 2019).

A promising approach for automated contouring of ROIs is deep learning (DL). DL is a subform of machine learning (ML). It uses neural networks, which are algorithms of a multi-layered structure, to analyze data and form conclusions. These neural networks are inspired by the human brain and learn from large amounts of data. Due to the availability of big data and enhanced computing power, DL has become popular for big data tasks such as speech and image recognition in recent years. Research has even shown that DL outperforms humans in visual and auditory recognition tasks (Lee et al., 2017), which is why it is explored in medical imaging as well. For biomedical image segmentation, the U-Net architecture (Ronneberger, Fischer, & Brox, 2015) and its variations are widely used.

In 2021 Sengun et al. conducted a comparative study about automatic liver segmentation in CT images (Sengun, Cetin, Guzel, Can, & Bostanci, 2021). They implemented a U-Net, Dense U-Net, and SegNet architecture and trained them on images and masks of 20 patients. Preprocessing of the data included Gaussian based smoothing and anisotropic diffusion filter methods to get rid of noise and enhance detail in low-resolution images. All networks received a mean Dice coefficient score of 0.7 or higher and SegNet was appointed as the best architecture, as it was more successful in eliminating false-positives and detecting positives more comprehensively.

While automatic liver segmentation has been proven to work well on CT and conventional MRI data, only a few studies can be found on automatic segmentation on qMRI data (Lind, 2017; Stacke, 2016). We hypothesize that segmentation on qMRI data can be automated with a U-Net and/or SegNet architecture. To investigate this, we have trained a U-Net, SegNet, and some variations on data of 37 patients with NAFLD and 15 healthy volunteers (52 in total). The performances of the models were compared afterwards.

CHAPTER 2

Background

2.1 Quantitative MRI

Quantitative MRI aims to extract quantifiable features from medical images, which can be used for e.g. assessing disease severity. qMRI methods require a series of images. In these images, a contrast setting of the MRI is varied, such as its sensitivity to diffusion, resulting in repeated measurements with different contrast weightings. By subsequently fitting a model that relates the contrast to the underlying physiology, it enables experts to directly probe and quantify microstructural properties (such as diffusion) of the tissues (Hall-Craggs, Bray, Ciurtin, & Bainbridge, 2019). So contrary to conventional MRI, which only provides anatomical information, qMRI provides extra information about the underlying tissue properties. This means that in qMRI, the intensity of each pixel in the image corresponds to a measurement of a real physical property.

Image segmentation or contouring is one of the most important tasks in medical image analysis and processing. Segmentation is crucial for facilitating the delineation, characterization, and visualization of ROIs in medical images, as well as surgical planning and image-guided interventions (Patil & Deore, 2013). Segmentation of qMRI images is crucial, because it gives an insight into the underlying tissue properties of an ROI instead of just qualitative information. For example, in researching the assessment of disease severity along the NAFLD spectrum, segmentation of the liver was crucial to access the diffusion properties of the liver (Troelstra et al., 2021). In their paper, Troelstra et al. research the use of qMRI data to find a correlation between tissue properties and disease activity as a replacement for performing a liver biopsy. They found that diffusion and perfusion significantly correlated with disease activity.

In conventional MRI, it typically takes 5 minutes to generate a single high-quality image, while qMRI requires 5-150 repeated images at different contrast weightings in a similar time. This drastically impacts image quality. Therefore, contouring on qMRI is not as trivial as it is for conventional MRI.

2.2 Diffusion Weighted Imaging

DWI is a mechanism for developing image contrast. It relies on changes in the diffusion properties of water molecules in tissues by using specific MRI sequences (Kele & van der Jagt, 2010). It provides qualitative and quantitative information about the diffusion properties, making it part of qMRI. Diffusion of water molecules is a physical property and it describes the Brownian motion, the random movement, of the water molecules. In tissues with low cellularity, water molecules can move freely. While in tissues with high cellularity, e.g. tumors, abscesses, fibrosis, and cytotoxic edema, the movement of water molecules is restricted (Kele & van der Jagt, 2010). Each tissue has unique diffusion properties, which makes it easy to distinguish healthy tissue

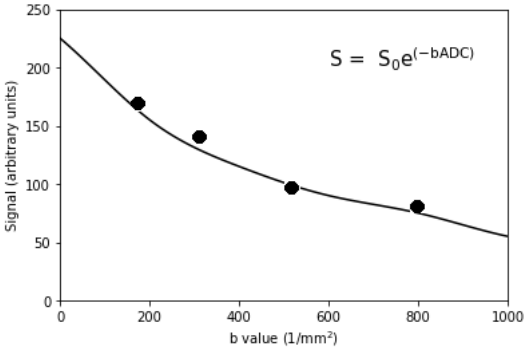


Figure 2.1: Signal Intensity in DWI (Jennings et al., 2002)

from unhealthy tissue by measuring the diffusion.

The b-value measures the degree of diffusion weighting that was applied in DWI. The higher the b-value, the more sensitive the sequence is to diffusion effects, the lower the intensity signal (see Figure 2.1). The signal intensity is determined by equation 2.1, where S_0 is the signal intensity without diffusion.

$$S = S_0 e^{-bADC} \tag{2.1}$$

In DWI, at least two b-values are used to measure the apparent diffusion coefficient (ADC). Using the signal intensities at different b values, the fitting technique described in section 2.3 is used to determine the ADC. Figure 2.1 is an example of such a fitted function. ADC Maps provide a non-invasive measure of cellularity, which makes DWI a useful tool in diagnosis and treatment planning and monitoring (Kele & van der Jagt, 2010).

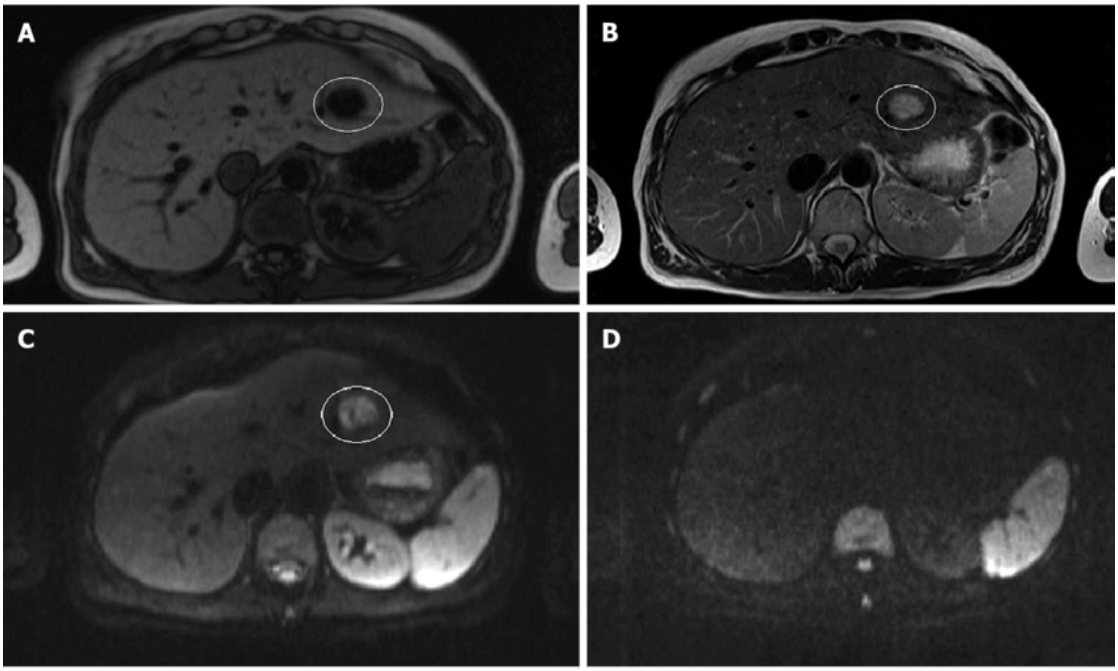


Figure 2.2: A: T1-weighted MRI; B: T2-weighted MRI; C: Diffusion weighted image (b-value 50 s/mm 2); D: Diffusion weighted image (b-value 1000 s/mm 2) (Kele & van der Jagt, 2010)

Figure 2.2 shows the difference between conventional MRI and qMRI. The figure depicts conventional T1- and T2-weighted images (A, B) and DWI images (C, D). Because the quality of the DWI images is not as good as that of conventional MRI images, qMRI is not as explored in the field of computer science and AI. However, AI algorithms that provide automatic segmentation of the liver could contribute to automatizing the pipeline of accessing quantitative information of a region of interest. This in turn could contribute to an optimized workflow in guiding diagnosis and treatment.

2.3 Deep Learning and Machine Learning in Medical Imaging

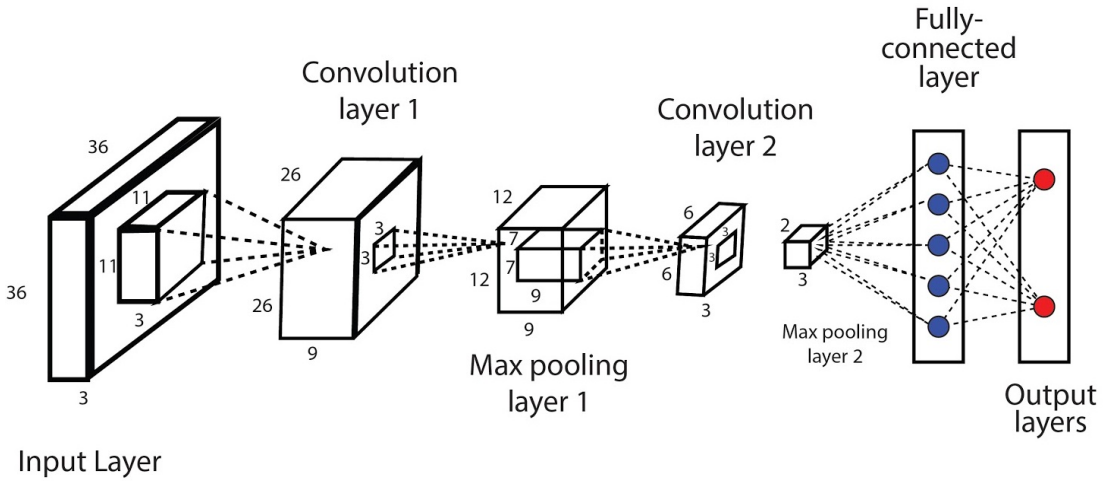


Figure 2.3: CNN Architecture (LeCun, Bengio, et al., 1995)

Deep learning is a sub-form of machine learning, which in turn is a sub-form of artificial intelligence. Artificial neural networks (ANNs) form the backbone of any DL or ML architecture. ANNs consist of a collection of nodes called (artificial) neurons, which are inspired by biological neurons. Each neuron processes incoming information, either from data input or from other neurons, and returns a weighted sum of outputs. Then, a bias is added to this sum to offset the final output. Finally, the computed sum is fed into an activation function which decides what is fired to the next neuron. The weights on each node are adjusted based on the error calculated on each forward propagation to maximize correct answers (Currie, Hawk, Rohren, Vial, & Klein, 2019). A deep neural network is an ANN that consists of many layers of neurons.

Convolutional neural networks (CNN), a class of deep neural networks, are most commonly used in image analysis. A CNN allows features to be extracted from images by using convolution and pooling layers (see Figure 2.3). The convolution layers extract features of the input images by using kernels. Kernels slide over the input images to produce a matrix of outputs, a feature map, that summarizes detected features in the input image. This is known as the convolution operation (see Figure 2.4). The max pooling layers typically follow after individual convolution layers in a CNN. The max pooling operation reduces the number of parameters by splitting up the image in 'pools' and extracting the maximum value from each pool, which leads to dimensionality reduction and easier computation.

The output of a CNN typically is some form of classification (Currie et al., 2019). Therefore, DL is being explored in medical imaging for purposes of diagnosis and detection of malignancies, segmentation of healthy tissue and malignancies, prediction of clinical outcome, and monitoring of treatment efficacy among other things (Boldrini et al., 2019). These DL techniques help ensure

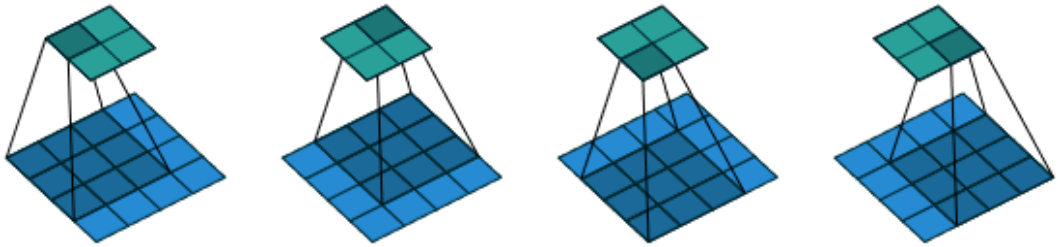


Figure 2.4: The convolution operation (Dumoulin & Visin, 2016)

contouring quality and consistency in case of image segmentation and reduce the time required for treatment planning, which contributes to an optimized workflow.

Manual segmentation is a tedious and repetitive task. Cardobi et al. showed that an AI algorithm can outperform manual segmentation in terms of time required. The mean manual segmentation time required to segment 39 pre- and post-chemotherapy CTs of the liver was 660 seconds, while the AI-assisted segmentation was 6.7 times faster with a mean segmentation time of 98 seconds (Cardobi et al., 2021). With help of current DL techniques, the workload and time required for manual segmentation can be lifted off of the shoulders of radiologists, while also gaining other benefits such as more consistent contouring quality.

2.3.1 Data Augmentation in Medical Imaging

One of the big problems in ML is the lack of sufficient training data. Medical data is less available for several reasons, including the fact that medical images are obtained less frequently than everyday images. Also, they are not often labeled and because of privacy issues, medical datasets often cannot be publicly published. Data augmentation is a promising technique to increase the amount of available training data. It is a technique where (random) alterations are applied to the dataset to enlarge it and to make the model more robust during training. For MRI images, data augmentations that have been proven to be useful are random rotation, crop and scale, and flipping (in images with symmetric autonomy) (Safdar, Alkobaisi, & Zahra, 2020). In these techniques, an original MRI image and its mask gets rotated, cropped and scaled, or flipped to produce extra training data. Less successful augmentation methods include adding noise because more variance occurs in the validation sets. Hao et al. worked on data augmentation strategies for diffusion-weighted MRI and supported the findings that were mentioned before (2020). In their research, the most useful data augmentation techniques turned out to be random rotation and translation (of motion).

2.3.2 U-Net for Medical Image Segmentation

In 2015, Ronneberger et al. proposed a CNN architecture called U-Net, with as main objective to include a localization output (2015) to a classic CNN output. The U-Net architecture contains an encoder and a decoder path (Ronneberger et al., 2015). The encoder path consists of convolutional and pooling layers and is used to capture context in the image. The decoder path is used to enable localization using upsampling. The localization part is what distinguishes the U-Net from a normal CNN. Building upon the U-Net architecture, Çiçek et al. proposed a 3D U-Net by replacing all 2D operations with their 3D counterparts (2016). The U-Net architecture, being designed for biomedical image classification, captures context as well as localization in images, making it possible to spot certain organs or tumors in a medical image. Therefore, the U-Net architecture is assumed to work well on liver segmentation as well.

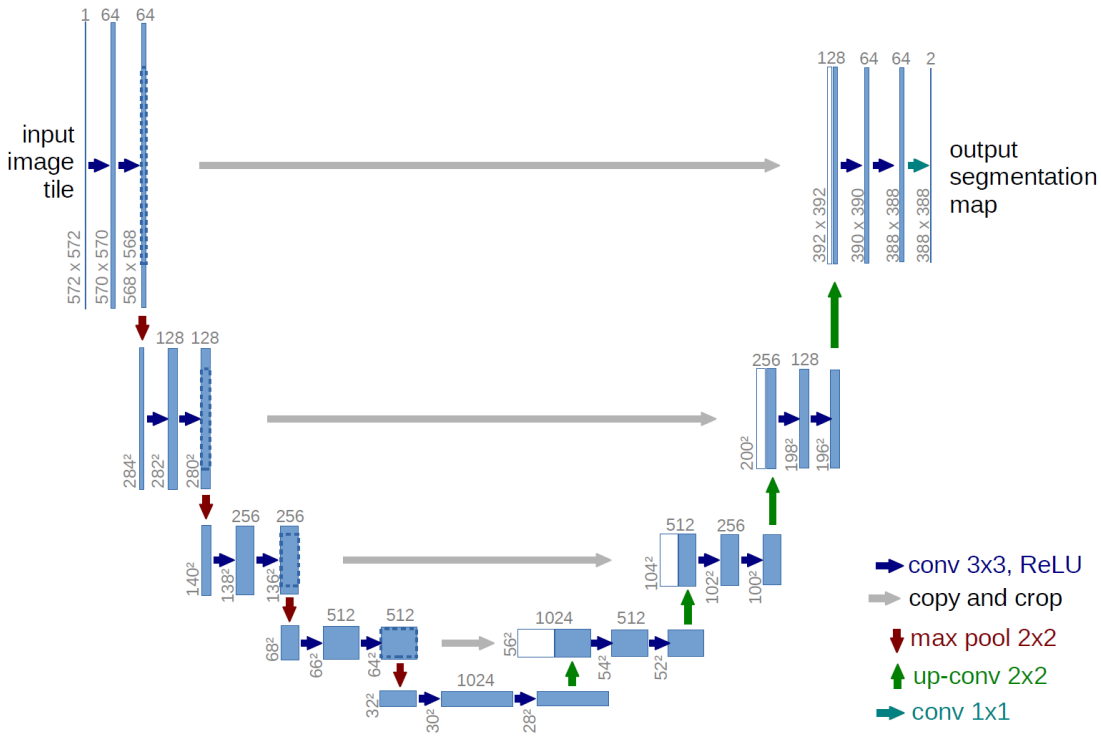


Figure 2.5: U-Net Model Architecture (Ronneberger et al., 2015)

The architecture of the U-Net model is illustrated in Figure 2.5. The U-like shape of the model is due to the encoder part of the model, the descending part in Figure 2.5, and the decoder part of the model, the ascending part in Figure 2.5. The encoder part consists of five blocks, each of which consists of two convolutional layers with a filter size of 3x3 and a ReLU activation function. The blocks get down-sampled four times using a max pooling layer with a stride of 2x2. The decoder part consists of four blocks, which in turn consist of the same convolutional layers as the blocks in the encoder path. They are up-sampled using up-convolutional layers with a stride of 2x2. Each decoding block concatenates the feature map produced from the decoding layer, this is called a skip-architecture. It is considered the key component of this architecture, as it combines the high-level representation (the 'what') from the decoding layers, with the appearance representation (the 'where') from the encoding layers to produce detailed segmentation (Dong, Yang, Liu, Mo, & Guo, 2017). In the final layer, a 1x1 convolution is used to map the final feature vector to a binary segmentation image. The network has 23 convolutional layers in total.

2.3.3 SegNet for Medical Image Segmentation

The SegNet architecture was proposed in 2017 by Badrinarayanan et al. The SegNet and U-Net architecture share a similar architecture, as they are both composed of an encoder- and decoder path. The main difference between the two architectures is that SegNet reuses pooling indices instead of transferring the entire feature map in the skipped connections (see 2.6), making SegNet computationally more efficient (Badrinarayanan et al., 2017).

The encoder part of the SegNet architecture consists of 13 convolutional layers which output feature maps. These feature maps are batch normalized, which is a technique to accelerate training en reduce generalization error, followed by a ReLU activation function. Lastly, a 2x2 max pooling operation with a stride of 2 is applied. The skipped connections serve the purpose of transferring boundary information in the encoder feature maps to the encoder. Instead of the whole feature map, they contain only the max-pooling indices, i.e., the locations of the maximum feature value in each pooling window. The decoder part of the architecture also consists of 13

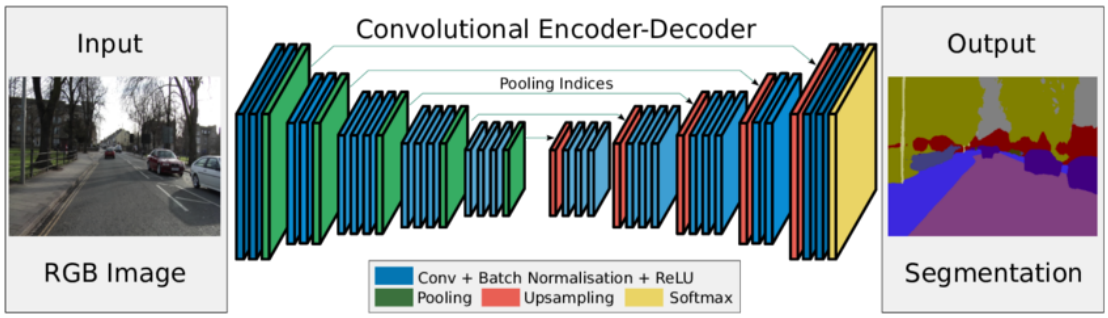


Figure 2.6: SegNet Model Architecture (Badrinarayanan, Kendall, & Cipolla, 2017)

convolutional layers. The input feature maps get upsampled using the pooling indices from the corresponding encoder feature maps.

SegNet was originally designed for scene segmentation, but has been proven to work well on biomedical image segmentation as well (Saood & Hatem, 2021; Alqazzaz, Sun, Yang, & Nokes, 2019).

2.4 K-fold cross-validation

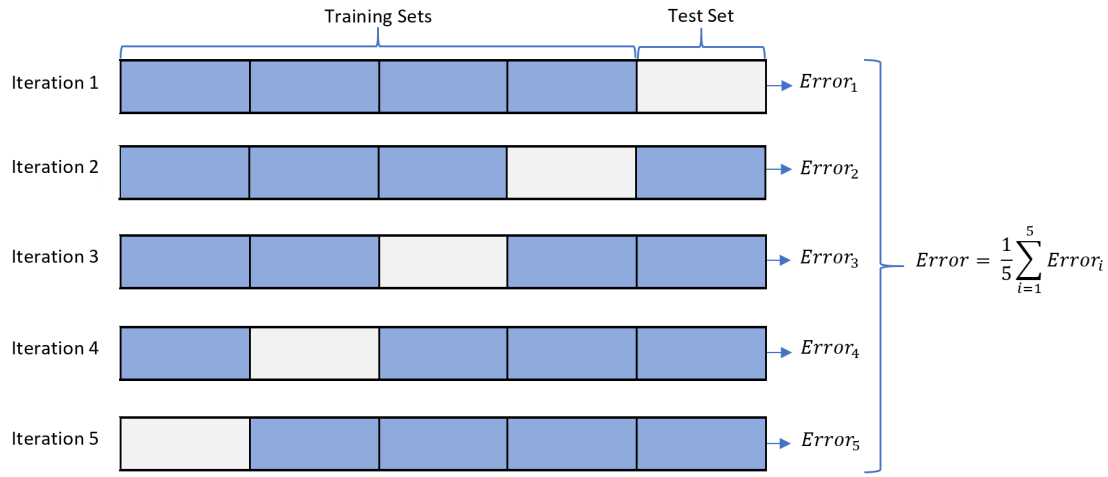


Figure 2.7: K-Fold cross-validation

In an ideal case for ML, an unlimited amount of data is available. Training a model on an infinite or large amount of data automatically ensures generalization. In a real-world setting, however, a limited amount of data is available. In such a setting, saving data for evaluation reduces the amount of data available for training. Therefore, to ensure sufficient data remains available for training, typically, 20% of the data is saved for evaluation. For a small dataset of e.g. 50 patients, this means that only 10 patients are reserved for testing. It is very well possible that these 10 patients are not representative of the general population. Disadvantages can arise when the test set includes outliers that the model is not used to. In this case, the accuracy scores of the model would be very low while it would perform well on normal data. It could also be the case that the test set includes data that is very similar to the training set, but not to real-world data.

In this case, the model would perform well on the test set, but not on real-world application data.

To avoid such disadvantages, k-fold cross-validation training can be used. It works as follows. First, the dataset is split up into k different configurations of a train, validation, and test set, as illustrated in Figure 2.7, called folds. Then, a model is trained on each of the folds and the error and/or accuracy for that fold is calculated. After that, the model is discarded. This is repeated for each of the k folds. Finally, the errors and/or accuracies for each fold are summed and the mean score is calculated. This gives an idea of how well the model performs in each of the k folds.

The advantage of this technique is that the complete dataset is considered for training as well as testing. This makes sure that outliers in the dataset have been considered, resulting in a generalizing model score. Furthermore, the training is stochastic and when training multiple times, this results in a mean training process. Because the models are discarded after each fold, this training technique does not result in one generalizing model but rather shows that the technique behind it can work well.

CHAPTER 3

Materials and Method

3.1 Data Description

The dataset used for this thesis was collected by the Department of Radiology and Nuclear Medicine at the AUMC (Amsterdam University Medical Centre). The dataset consisted of qMRI (DWI) images of the abdominal area of 37 patients with NAFLD and 15 healthy volunteers, making it a total of 52 subjects. Per subject, each image contained 27 slices. This means that the dataset contained $27 * 52 = 1404$ 2D images. A professional radiologist annotated two corresponding masks: one where the whole liver is annotated and one where the liver, excluding the blood vessels, is annotated. See Figure 3.1 for an example of a data slice and the corresponding two masks.

Figure 3.1: Example image slice and corresponding masks

The data is in the Neuroimaging Informatics Technology Initiative (NIfTI) format (Cox et al., 2004). This file format was developed in the early 2000s and is still used nowadays for medical imaging and radiology research. Our NIfTI files have the following dimension format: $[x, y, z, b]$, where x, y and z define the three spatial dimensions and b defines the b-value, which measures the degree of diffusion weighting that was applied (see equation 2.1).

3.2 Data Processing

All (pre)processing steps in this project were conducted with the python package Medicaltorch (Perone, cclauss, Saravia, Ballester, & MohitTare, 2018). Medicaltorch is an open-source framework that builds upon PyTorch (Paszke et al., 2019), providing an extensive set of loaders, pre-processors, and datasets for medical imaging. The source code of this package was adapted

to fit the dataset where needed. Using the loaders provided by Medicaltorch, the (pre)processing steps were done 'on the fly' while the data was loaded into the model.

3.2.1 Slice Filtering

A slice filter was applied to the data to bypass data slices that did not contain useful information. Masks that did not contain an annotated liver, were removed from the data. The reasoning behind this approach is the following. In these completely black masks, the liver was partially visible but not annotated. When the model was trained, it recognized the part of the liver that was visible and gave back an annotated output, which was then punished by the loss function because it did not overlap with the ground truth mask. The resulting score of the model would not be accurate, because the model was not particularly wrong in this situation. As a result of removing the slices that did not contain any ground truth masks, 480 slices were removed The remaining slices all contained annotated masks.

3.2.2 Data Augmentations

All images and the corresponding masks had a shape of 256 x 256 voxels. For training, k-fold cross-validation was used (see section 2.4), resulting in a train-, validation- and test set per fold. The data was split on patient level. Data augmentations were applied to the training sets to increase the number of training images. Per fold, 80% of the training data was augmented, while 20% of the data was left as is. Figure 3.2 shows the augmentations that were applied in the middle row. The last row, which shows extreme augmentations, has been added to show the importance of using small augmentation ranges to make sure the augmented data still is representative of real-life data. The following sub-sections will touch upon the applied augmentations and how they have been used.

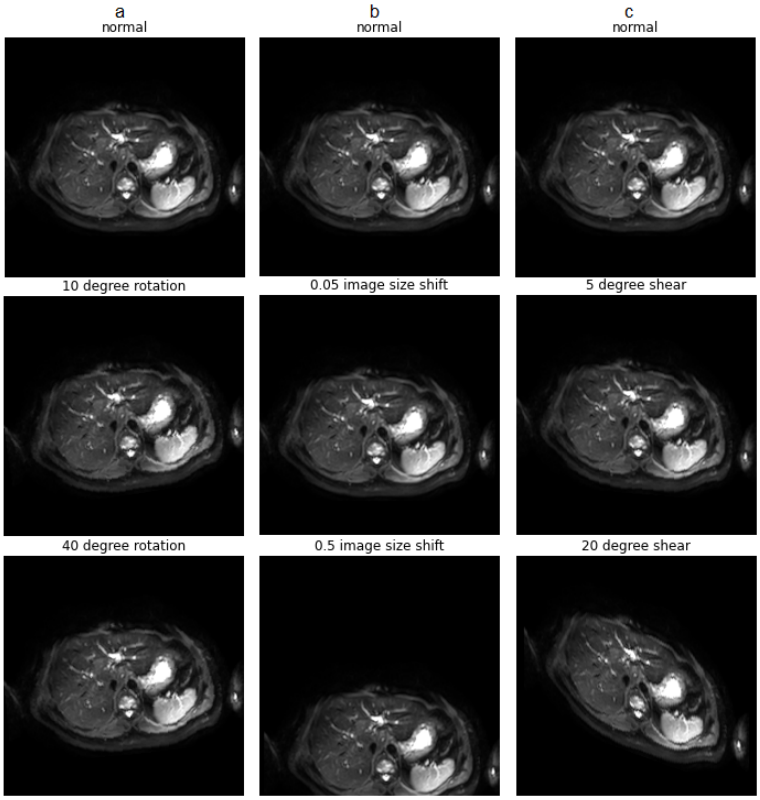


Figure 3.2: Image augmentations, from top to bottom: normal, applied augmentation, extreme augmentation

Rotation

The dataset that was used for this project contains images of the abdominal area, including the liver. In MRI scans, the liver is always on the same side. Therefore, only slight rotations are representative of real-life data. For example, a 90 or 180-degree rotation would not make sense because the liver would never be flipped in an actual MRI scan. Considering this, a random rotation of maximum 10 degrees was applied to the training data. This indicates that the data was rotated by a maximum of 10 degrees in a random direction as can be seen in Figure 3.2a. Note that the augmentations were also applied to the masks with threshold interpolation, even though they are not depicted in the figure. Every pixel lower than 0.5 was rounded to 0, and every pixel higher than 0.5 was rounded to 1. This is the case for every depicted augmentation.

Shift

Furthermore, a shift or translation of maximally 0.05 times the image size was applied to the training data. This means that the data was shifted from its place by a small factor of 0.05 in the up-, down-, right- or leftward direction. This is again to reproduce the most life-like augmented images, as there is not much room to move in an actual MRI scan. An example can be seen in Figure 3.2b.

Shear

Shearing an image means to shift a part of the image in one direction while shifting the other part to another direction. This creates a parallelogram effect on the image, as the shape of the image is transformed into a more skewed shape. The shear was applied in the shear range of -5,5 degrees, resulting in a shear in the up-, down-, right- or leftward direction. An example can be seen in Figure 3.2d.

3.2.3 Histogram Clipping and Range Normalization

Histogram clipping was performed on the top and bottom 1% of the intensity values, resulting in more normalized intensity values (because the possible outliers are cropped away) and contrast enhancement, clearly visible in Figure 3.3 below.

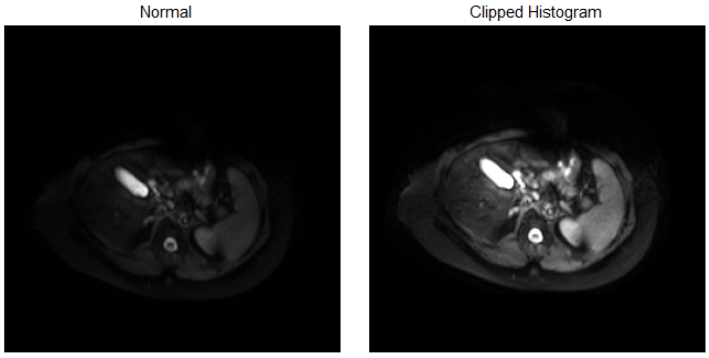


Figure 3.3: The result of Histogram Clipping

Range normalization was applied to the images to bypass intensity variation caused during MR image acquisition. The values were normalized to lie between a value of 0 and 255 (see Figure 3.3, right), while prior to range normalization the intensity values reached up to a value of 600 or more (see Figure 3.3, left).

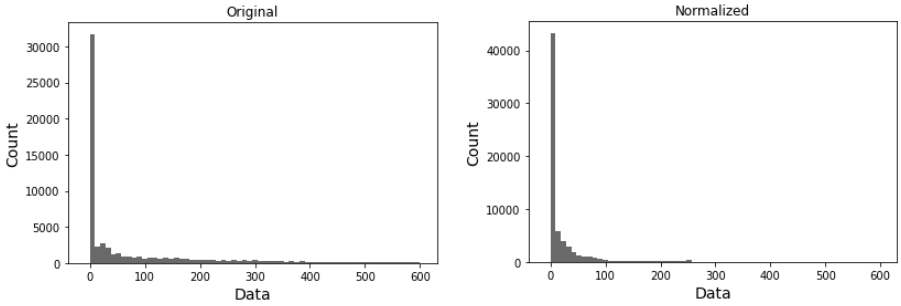


Figure 3.4: Original versus normalized histogram

3.2.4 post-processing

The final step in the data processing pipeline was the post-processing of the predicted masks. Per NiFTI file, the predictions were done per slice. All predicted slices were then stacked along the third dimension, the first two being height and width. After the predicted slices were stacked, the slices were converted into the NiFTI file format using the NiBabel package (Brett et al., 2020), which is a python package that provides read and write access to common medical and neuroimaging file formats. The newly generated NiFTI file could then be opened with visualization tools that support this format, such as FSLeyes (McCarthy, 2021). The image below provides an example of the visualization of a qMRI NiFTI file, with the predicted mask overlayed on top.

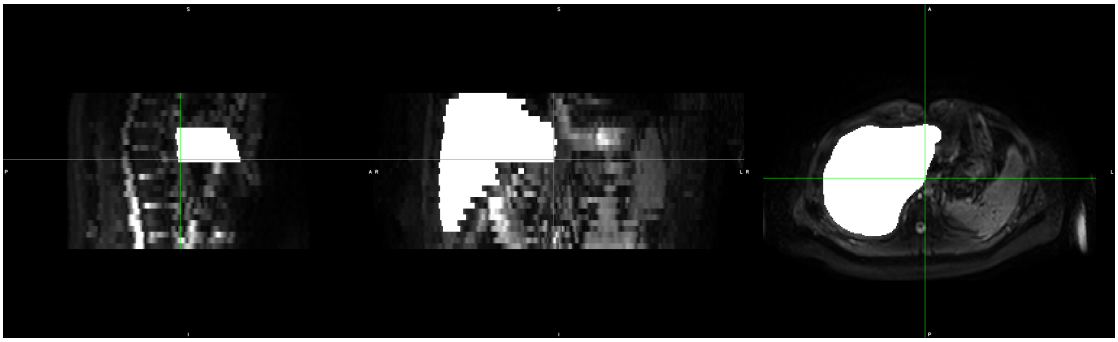


Figure 3.5: qMRI image with corresponding predicted mask for the liver opened in FSLeyes

For slice level post-processing, a Canny edge detector (Canny, 1986) was applied to the predicted segmentation masks to obtain a delineation of the predicted liver. In this way, the delineation could be overlapped with the qMRI image, illustrated in Figure 3.6.

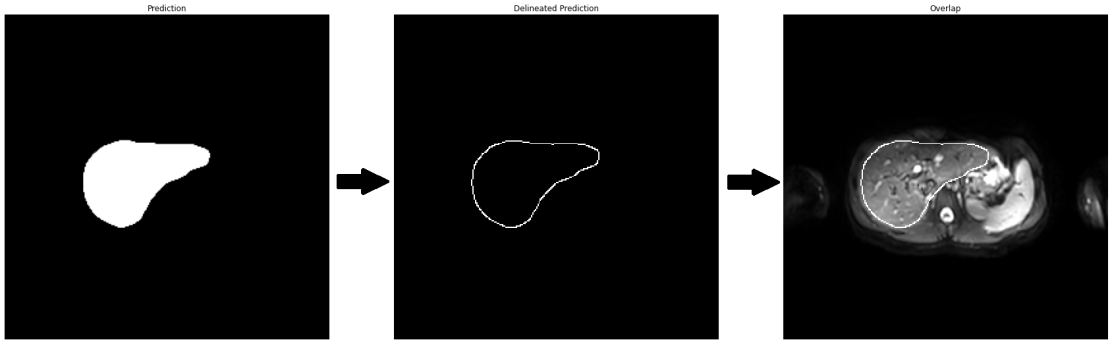


Figure 3.6: Segmentation delineation using Canny edge detector

3.3 Experiment Details

3.3.1 Data Handling

Distribution	**%**	**Subjects**	**2D Slices**
Train	70	37	647
Validation	10	5	92
Test	20	11	185

Table 3.1: Data distribution

For training, a 5-fold cross-validation training loop was used (see section 2.4). Per fold, the data was divided into train-, validation- and test sets. The splits were the same for each fold. The specific distribution is set out in Table 3.1. The data split was performed at patient level, instead of at slice level, to make sure that the data of one patient was not scattered over multiple sets, as this would introduce a bias. Namely, if different images of the same patient are present in the train set as well as the test set, the network has a higher chance of getting the test images correct, even though the images are not the same. This is the case because the network has already learned about the structure of that particular subject, hence the bias. The validation set was used for early stopping. A counter was implemented that kept track of the number of times the validation loss did not decrease. After 25 times without decrease, the training was stopped and the best model was saved.

The data was loaded into the model using an MRI dataset class from the MedicalTorch package (Perone et al., 2018) and a custom Pytorch data loader. The data was shuffled inside the data loader. The dataset class has been adapted for three more functions: multi-channel training, average image training, and transfer learning, which will be explained in section 3.4.

3.3.2 Hyperparameters

Parameter	**Value**
Epochs	max 250
Batch Size	16
Learning Rate	0.001
Optimizer	*adam*
Folds	5
Loss Function	*Dice loss*

Table 3.2: Training Hyperparameters

All implemented architectures were trained for 5 folds, with a maximum of 250 epochs and early stopping, on an Nvidia Tesla P100-PCIE GPU. The hyperparameters used for training were the following. A batch size of 16, a learning rate of 0.001, an adam optimizer (Kingma & Ba, 2014) and a Dice loss (Sørensen, 1948; Dice, 1945) loss function (see Table 3.2). The Dice loss function computes the overlap between the ground truth and the mask image. It is also used as the evaluation metric, it will be further explained in section 3.5. These configurations were heuristically optimized, combined with literature (Ronneberger et al., 2015). They were also used in training with the mask without veins (Figure 3.1, right).

3.4 Architectures and Approaches

In addition to the conventional U-Net model and input handling (section 2.3.2), two more architecture functionalities and one additional approach were implemented. They are explained in

the following subsections, along with the conventional models that were used.

3.4.1 U-Net

The conventional U-Net model (see section 2.3.2) was trained with the hyperparameters in Table 3.2. This model was particularly designed for biomedical image segmentation (Ronneberger et al., 2015) and has been proven to work well for liver segmentation on conventional CT and MRI data (Sengun et al., 2021; Frid-Adar, Klang, Amitai, Goldberger, & Greenspan, 2018). Even though qMRI data differs from conventional MRI data, e.g. in image quality, the U-Net model was chosen for this project because of its skip-architecture (see 2.5) which enables capturing context as well as localization.

3.4.2 SegNet

The conventional SegNet model (see section 2.3.3) was trained with the hyperparameters in Table 3.2. Originally designed for scene segmentation, this model was chosen because it was proven to work well on biomedical image segmentation and because it is computationally more efficient than the U-Net model (Sengun et al., 2021; Saood & Hatem, 2021; Alqazzaz et al., 2019; Badrinarayanan et al., 2017).

3.4.3 Multi-Channel U-Net

The conventional U-Net model (see section 2.3.2) that was used for this project expects an input of one NIfTI file, consisting of one channel (because the data is black and white, instead of having color). In the multi-channel training implementation, the input channel of the U-Net changes from one input channel to two input channels. The total input that is fed into the model consists of two images with different b-values, instead of just one image with one b-value. The idea behind this approach is that the network gets different views of the diffusion in the liver at different b-values so that it can better distinguish the liver. Especially for detecting the liver without the veins, this is thought to be helpful, as the intensity of the veins changes with different b-values. The b-values used here were 0 (no diffusion) and 700 (high diffusion) because they differ strongly. Note that two or more channels could be inputted with this approach, but two channels were chosen for simplicity purposes.

3.4.4 Average Image U-Net

In the average image training implementation, an average image is calculated from multiple images of the same b-value. This results in one NiFTI file input of one channel. Using an average image can help with correcting movement distortions in the MRI images because the movements get averaged.

3.4.5 Transfer Learning

The transfer learning approach is focused on the vessel mask (see Figure 3.1, right). In this approach, the U-Net model was first trained on the whole liver mask (see Figure 3.1, middle). The model weights were saved and then used again to train the network further on the vessel mask. The idea behind this approach is that the model first learns to locate the liver so that it can focus on the veins in the second training process.

3.5 Evaluation Metrics

3.5.1 Dice loss

The evaluation metric that was used for this project is the Sørensen-Dice similarity coefficient (Sørensen, 1948; Dice, 1945), which is a popular metric to evaluate the similarity between two samples in image segmentation (also called Dice score). It is calculated using formula 5.1.

$$Dice(X,Y) = \frac{2|X \cap Y|}{|X| \cup |Y|} = \frac{2 \times TP}{(TP + FP)(TP + FN) + \delta} \tag{3.1}$$

The Dice score calculates the overlap between the ground truth and the predicted mask, which can be seen as the number of true positives. This is divided by the total number of pixels in both masks and multiplied by two. The δ is added in the denominator to prevent limit issues. The Dice score penalizes for the false positives (the pixels that have been incorrectly classified as the liver) and the false negatives (the pixels of the liver that the model could not find). This makes it a reliable metric for image segmentation because it deals with size biases. For example, when the number of white pixels in the ground truth masks are small in comparison to the black pixels in the background, this introduces a size bias because the white pixels are less present than the black pixels. In this scenario, the model would benefit more from returning a completely black prediction. The dice score penalizes that and only looks at the overlap between two segmentations instead of the amount of black and white pixels. The Dice score gives back a value between 0, indicating no similarity, and 1, indicating high similarity.

3.5.2 Paired Student's t-test

Because we use a k-fold cross-validation training loop, the data distribution is changed per fold. Changing the train-, validation- and test sets might change the model performance (Student, 1908). Therefore, to statistically compare the performance of the models, alongside comparing the Dice scores, a paired t-test, proposed by Gosset, under the pseudonym 'Student', (1908), was conducted on the results of the models. The null hypothesis of this test states that two models perform the same. The alternative hypothesis states that the models perform differently. To reject the null hypothesis, in favor of the alternative hypothesis, a $Pvalue$ smaller than the considered significance level (5%) must be found. In the case of a paired t-test, the $Pvalue$ is determined according to formula 3.2. Where TS stands for the test statistic and cdf stands for the cumulative distribution function of the distribution of the test statistic under the null hypothesis.

$$Pvalue = 200 * (1 - cdf(|TS|)) \tag{3.2}$$

The TS, as proposed by Gosset (1908), is calculated using the following formulas. Where n stands for the total number of obtained scores, $\bar{d}$ for the difference of the scores, and s_d for the standard deviation of the scores.

$$SE(\bar{d}) = \frac{s_d}{\sqrt{n}} \tag{3.3}$$

$$TS = \frac{\bar{d}}{SE(\bar{d})} \tag{3.4}$$

A $Pvalue$ lower than 5% would indicate that two ML models perform differently, meaning that the estimated difference in score is likely to be truly different instead of difference due to chance. A $Pvalue$ higher than 5% would indicate that the difference in the performance of the models was due to chance. However, a $Pvalue$ higher than 5% could also be obtained by having too few samples. Since we trained with a 5-fold cross-validation ($n = 5$), this was the case. Therefore, it was decided that for this research a $Pvalue$ lower than 10% is also acceptable. This would mean that there is a 90% chance that the models differ statistically, instead of due to chance.

CHAPTER 4

Results

	U-net	SegNet	MultiChannel	Average Image
Fold 1	0.9261	0.7459	0.9258	0.9175
Fold 2	0.9197	0.7612	0.9129	0.9116
Fold 3	0.8749	0.8657	0.8757	0.8671
Fold 4	0.9331	0.8440	0.9232	0.9267
Fold 5	0.9128	0.8730	0.9148	0.9052
Average	**0.9133***	**0.8179**	**0.9105**	**0.9056**

Table 4.1: Results Whole Liver Mask (Dice Score), * indicates best result

	U-net	SegNet	MultiChannel	Average Image	Transfer Learning
Fold 1	0.8634	0.8034	0.8532	0.8153	0.8566
Fold 2	0.8570	0.8145	0.8504	0.8328	0.8552
Fold 3	0.7945	0.7923	0.8010	0.7580	0.8104
Fold 4	0.8580	0.8211	0.8515	0.8342	0.8638
Fold 5	0.8528	0.7282	0.8499	0.8101	0.8559
Average	**0.8451**	**0.7919**	**0.8412**	**0.8101**	**0.8484***

Table 4.2: Results Vessel Mask (Dice Score), * indicates best result

	U-Net/SegNet	U-Net/TransferL
Pvalue Whole Liver Mask	0.0104	-
Pvalue Mask w/o Veins	0.0713	0.3883

Table 4.3: Paired t-test result from comparing the U-Net to other approaches

Tables 4.1 and 4.2 show the Dice score results of both masks in combination with all trained network architectures. The transfer learning approach is only applied to the mask without veins, hence it is not included in Table 4.1. We observe the following trends. Firstly, we see that the U-Net architecture performs better than the SegNet architecture. This is supported by the *Pvalues* found in conducting the Student's paired t-test (Student, 1908) on the U-Net and SegNet results. Both *Pvalue* (see Table 4.3) satisfy the condition $Pvalue<0.10$, as determined in section 3.5. Secondly, the multi-channel U-Net scores are very close to the conventional U-Net scores, while not having been trained with the same augmentations. The average image U-Net does not better the performance of the conventional U-Net. Furthermore, the transfer learning approach improves the scores of the conventional U-Net, but not significantly and statistically,

as its $Pvalue$>0.10 (see Table 3.2). Lastly, we see that all models perform worse or worst in fold 3.

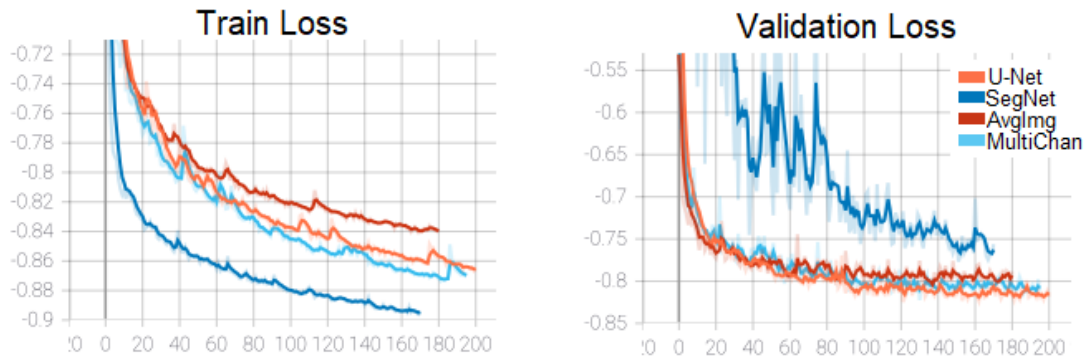


Figure 4.1: Train and Validation Loss, Fold 1 (vessel mask)

In Figure 4.1 we illustrate the train- and validation loss of the first fold in the training process of the vessel mask. The patterns observed here are also observed in the whole liver mask. In the training loss, we observe that the SegNet model decreases the most. The training loss of the other models decreases more slowly, with the multi-channel U-Net on top. In the validation loss, however, we see that SegNet has some troubles in the first 80 epochs, and stabilizes around 100 epochs. The loss of other models decreases faster, in a similar way to each other. In the validation loss, the U-Net channel eventually decreases the most.

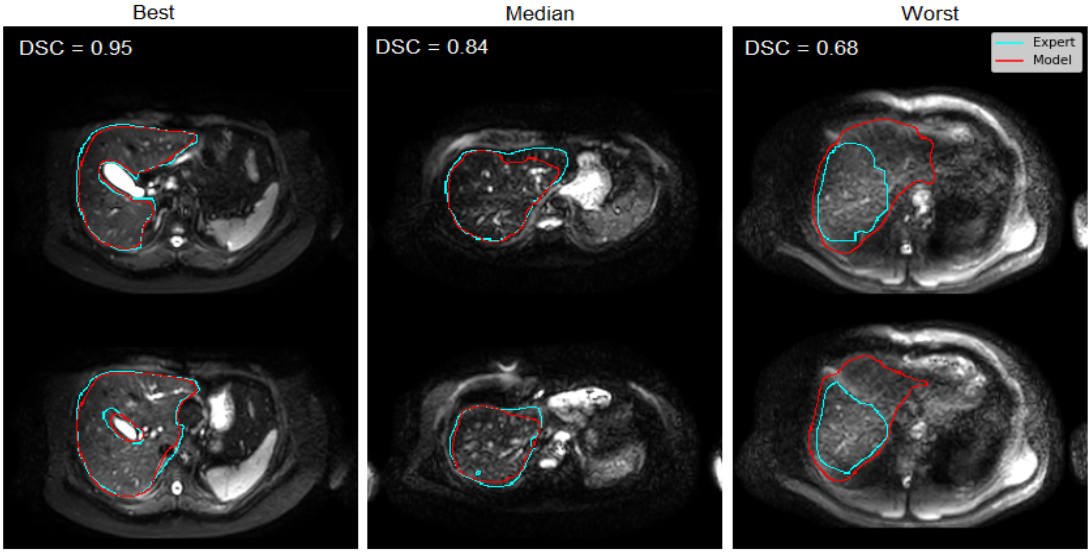


Figure 4.2: U-Net Results

An illustration of the best, median, and worst case results of the U-Net model can be seen in Figure 4.2. The light blue lines depict the segmentation of an expert, and the red lines depict the segmentation of the trained model. In the best case result, we see that the two lines overlap almost perfectly, even in the complicated structures (for example the highlighted gal bladder in the middle of the liver). The median results show a great overlap in segmentation and only a small variety in the exact shape of the liver. In the worst case results, the model predicts quite a bigger region in comparison to the expert.

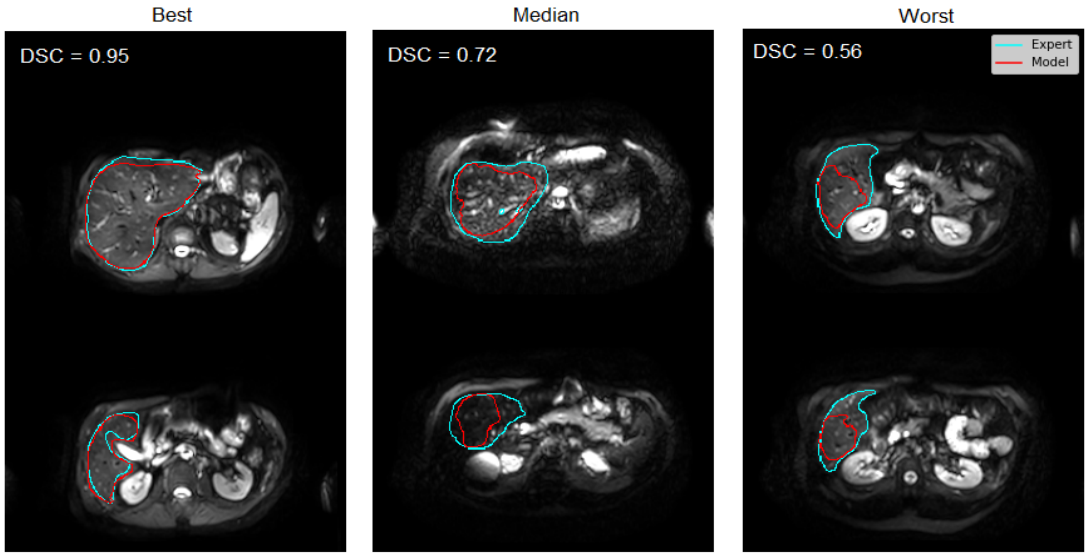


Figure 4.3: SegNet Results

Figure 4.3 shows the best, median, and worst case results of the SegNet model. The bast case result has an almost perfect overlap, just like the U-Net best case result. In the median and worst case results, however, the SegNet model seems to systematically predict smaller regions in comparison with the expert.

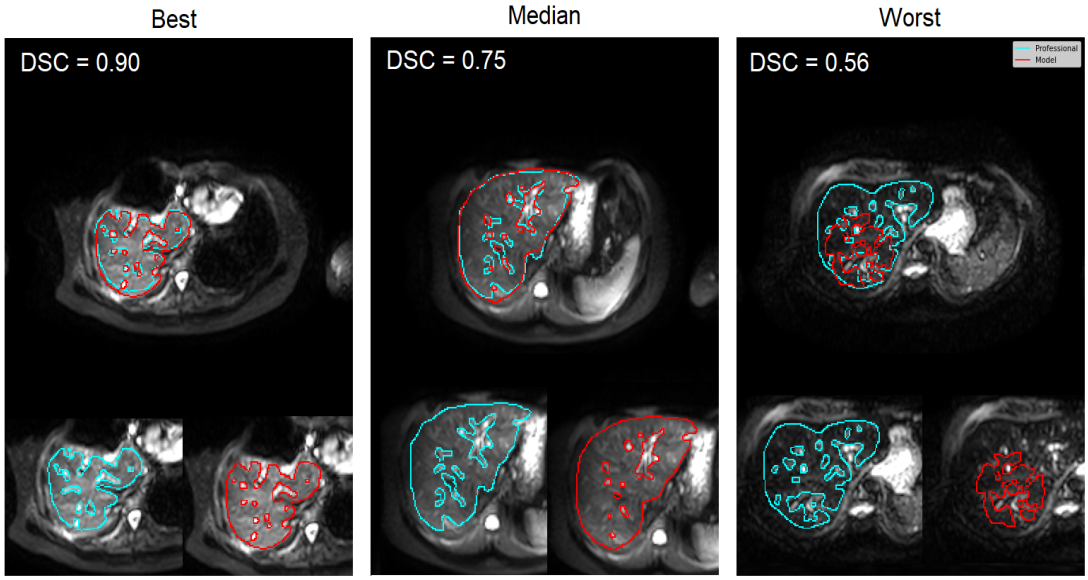


Figure 4.4: Transfer Learning U-Net Results (vessel mask)

As the transfer learning approach turned out to have the best Dice score for the vessel mask in this research, it is used to show the best, median, and worst case result of the vessel mask in Figure 4.4. In the case of the best and median results, we see that the delineation of the shape of the liver has almost a perfect overlap with the ground truth delineation. In the worst case result, the model is quite far off, predicting a complex shape around the highlighted parts of the image

slice. Also notable is that, while the model seems to be good in predicting the locations for the veins, it does not seem to get the shape of the veins right. The model predicts more bubbly and circular shapes, instead of the more complex delineations drawn by the expert.

CHAPTER 5

Discussion

The results show that the U-Net model performs better than the other models for both the whole liver mask and the vessel mask (see Table 4.1, 4.2). The SegNet model probably performs worse because it tends to make quite small predictions (see Figure 4.3). Of course, the difference in overlap between the ground truth and prediction is deduced from the Dice score, and if the model keeps making smaller predictions, this will add up to give a lower Dice score. It was noted before that the multi-channel U-Net results are very close to the conventional U-Net results, even though they have not been trained with augmentations, for both liver masks. This means that the networks indeed learn about the shape and location of the liver from multiple input dimensions with different b-values. The average image U-Net probably does not add much to the score for both masks because it does not change the images drastically for the model to learn new information.

Furthermore it was expected that the Dice scores of the vessel mask would be lower, as the structures are more complex and the veins are not always in a particular area. The transfer learning approach did not improve the score statistically, but it did improve convergence time. When the vessel mask was trained on the conventional U-Net model, the model converged to a maximum after 210 epochs. This was determined with the early stopping counter (see further explanation in section 3.3). While using the transfer learning approach, the model converged to a maximum in 140 epochs.

Even though the Dice score of the vessel mask was lover than the Dice score of the whole liver mask, the scores were still quite high. We saw that the best performing model had less trouble in predicting the liver shape (hence the high score), and more trouble in predicting the vein shapes. This could be explained by the following. By using the Dice loss in the model training, the model tries to maximize the overlap between the ground truth and predicted mask. Because the veins are small, they take up relatively little volume and do little on the loss. This makes it safer for the model to predict the veins as small as possible or not at all.

The worst case performance of the U-Net, seen in Figure 4.2 on the right, is an example from the images of the test set on which the model was validated in fold 3. We saw earlier that all models performed worse or worst in fold 3 (Table 4.1, 4.2). The worst case result in Figure 4.2 is an example of an image slice in fold 3. On the left side of the image, you can see that a sort of shadow, or darkness, covers the top left side of the abdominal area. It could be that this confused the model and that therefore the scores were somewhat more off than in the other folds. But when we look at the expert versus model segmentation, not only the liver shape differs quite a bit, but also the size. Another explanation as to why the scores in this particular fold were more off than the others is that the model performs better than the expert in this case. Literature has shown that human annotators are prone to error (Wang et al., 2019), and when looking at the picture intuitively, we see that the model takes the whole liver into account while the expert excluded the liver boundaries. The liver boundaries were excluded by the expert because she did

not trust those regions of the liver due to motion and poor image quality.

In comparison to the results of Sengun et al., who received a mean Dice score of 0.7 or higher, our results are higher. This could be explained by the fact that Sengun et al. did not use augmentations for their training, while our experiments did. Another explanation could be that our dataset consisted of 52 subjects, while theirs consisted of 20 subjects, giving our model the chance to generalize better due to having more data. Our obtained scores do however conform to other researches on automatic liver segmentation using deep learning (Wang et al., 2019).

5.1 Limitations

The limitations and their possible effects in this work are discussed in the following sections.

In the slice filtering pre-processing step, all image slices that did not contain a mask were removed from the dataset (see section 3.2.1). A limitation of this approach is that slices that do not contain the liver are removed so that the networks are not trained on cases where the liver is not visible. A result of this limitation is that it is unclear how the networks behave when they encounter an image slice that does not contain a liver.

Another limitation is that we did not take into account the reasons as to why the expert did not annotate certain image slices. Reasons could be, for example, because of motion distortion or artifacts. These reasons could have been taken into account in the multi-channel U-Net architecture. If the network received multiple inputs of different b-values, it could have potentially included movements better because it received more information. The artifacts could have possibly been detected by a 3D U-Net structure, as this also takes the spatial dimension into account, making it able to spot volumes that differ from the liver more easily.

Furthermore, for the multi-channel U-Net, a b-value of 0 and 700 was chosen for the input channels, as these values differ the most from each other. The idea was that the network gets extra information. However, in retrospect, a b-value of 700 might not be optimal to input in the network as the intensity signal in the liver is considerably less at that point. A better approach would be to use a b-value of 10 or 20 for the second input channel. The intensity signal in the liver at those points is higher, providing the image slices with more information. Even better would be to input three or four b-values to see if, and in what way, that would improve the network.

Lastly, the hyperparameters for training were heuristically optimized, instead of systematically. Systematical hyperparameter tuning leads to a reduction in training time and an improvement of performance (Smith, 2018). It is advised for optimal performance of classification tasks.

5.2 Directions for Future Work

To improve this work, we propose the following approaches.

An obvious approach is to use a 3D or 2.5D U-Net, instead of a 2D conventional U-Net. The liver is a 3D structure, and by training the data in slices on a 2D network, the spatial dimension is lost. In a 3D model, such as the one proposed by Çiçek et al., the input data can be three-dimensional and the model learns from the spatial dimension as well. However, training 3D data on a complicated network is time-consuming and requires high computational resources, which is why a 2.5D network was proposed for biomedical image segmentation (Wardhana, Naghibi, Sirmacek, & Abayazid, 2021). This 2.5D U-Net has a deeper and wider architecture in comparison to the conventional U-Net but saves computational costs in comparison to the 3D U-Net.

It was noted previously that the multi-channel U-Net architecture results were similar to the conventional U-Net architecture results, despite not having been trained with augmentations. Therefore we think that by training the multi-channel U-Net with augmentations the scores could outperform the scores of the conventional U-net. The augmentations would have to be implemented for a multi-channel input for this approach. Also, from the results (Table 4.1, 4.2) it was clear that the average image U-Net architecture did not improve the conventional U-Net architecture score. However, we propose that combining this average image approach with other approaches, e.g. the multi-channel approach, could contribute to achieving a higher score. Adding the average image approach could account for generalization in more complex models, as it takes an average of multiple images of the same b-values.

Another approach is to use Generative Adversarial Networks (GANs) for synthetic data augmentation. GANs automatically discover and learn the patterns of their input image data in such a way that they can be used to generate new images that resemble the input data (Frid-Adar et al., 2018). It is therefore that they have been adapted into frameworks for generating high-quality realistic and natural images in medical imaging. Using GANs for synthetic data augmentations in medical image segmentation has lead to an increase in performance (Frid-Adar et al., 2018), which is why it is proposed as a future step to improve this work.

To better the performance of the models on the vessel mask, we propose the following method. First, a U-Net architecture could be used to predict the liver in qMRI data, just like this project. Secondly, another U-Net architecture could be used to train and predict only the veins of the liver in the same dataset. Then, these networks could be combined to achieve a segmentation of the liver, excluding the veins, by subtracting the vein predictions from the liver predictions.

CHAPTER 6

Conclusion

For this project, we hypothesized that liver segmentation on qMRI data could be automated with a U-Net and SegNet architecture. To investigate this, we trained a U-Net, SegNet, and some variations on image data and two corresponding masks of 37 patients with NAFLD and 15 healthy volunteers (52 in total). For the mask where the whole liver was annotated all models scored above a 0.8 Dice score. The best performing model was the standard U-Net architecture, which received a mean Dice score of 0.91. This conforms to state-of-the-art literature about deep learning in automatic liver segmentation on CT and MRI image data (Wang et al., 2019). The vessel mask performed worse which was anticipated because of the more complex structure. To improve scores on the vessel mask, we propose to train two models for segmenting the liver and the veins separately and extracting them from each other. Our results were higher than those of Sengun et al., probably because of the successful image augmentations that were applied and the larger dataset. With this, we proved that automatic liver segmentation on qMRI data can indeed be automated with a U-Net and SegNet architecture.

References

Ahmed, M. (2015). Non-alcoholic fatty liver disease in 2015. *World journal of hepatology*, *7*(11), 1450.

Alqazzaz, S., Sun, X., Yang, X., & Nokes, L. (2019). Automated brain tumor segmentation on multi-modal mr image using segnet. *Computational Visual Media*, *5*(2), 209–219.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *39*(12), 2481–2495.

Boldrini, L., Bibault, J.-E., Masciocchi, C., Shen, Y., & Bittner, M.-I. (2019). Deep learning: a review for the radiation oncologist. *Frontiers in oncology*, *9*, 977.

Brett, M., Markiewicz, C. J., Hanke, M., Côté, M.-A., Cipollini, B., McCarthy, P., ... freec84 (2020, November). *nipy/nibabel: 3.2.1.* Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.4295521` doi: 10.5281/zenodo.4295521

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*(6), 679–698.

Cardobi, N., Dal Palù, A., Pedrini, F., Beleù, A., Nocini, R., De Robertis, R., ... D'Onofrio, M. (2021). An overview of artificial intelligence applications in liver and pancreatic imaging. *Cancers*, *13*(9), 2162.

Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016). 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention* (pp. 424–432).

Cox, R., Ashburner, J., Breman, H., Fissell, K., Haselgrove, C., Holmes, C., ... Strother, S. (2004, 01). A (sort of) new image data format standard: Nifti-1. *10th Annual Meeting of the Organization for Human Brain Mapping*, *22*.

Currie, G., Hawk, K. E., Rohren, E., Vial, A., & Klein, R. (2019). Machine learning and deep learning in medical imaging: intelligent imaging. *Journal of medical imaging and radiation sciences*, *50*(4), 477–487.

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

Dong, H., Yang, G., Liu, F., Mo, Y., & Guo, Y. (2017). Automatic brain tumor detection and segmentation using u-net based fully convolutional networks. In *annual conference on medical image understanding and analysis* (pp. 506–517).

Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*.

Frid-Adar, M., Klang, E., Amitai, M., Goldberger, J., & Greenspan, H. (2018). Synthetic data augmentation using gan for improved liver lesion classification. In *2018 ieee 15th international symposium on biomedical imaging (isbi 2018)* (pp. 289–293).

Hall-Craggs, M. A., Bray, T. J. P., Ciurtin, C., & Bainbridge, A. (2019). Quantitative magnetic resonance imaging has potential for assessment of spondyloarthritis: arguments for its study and use. *The Journal of rheumatology*, *46*(5), 541–542.

Hao, R., Namdar, K., Liu, L., Haider, M. A., & Khalvati, F. (2020). A comprehensive study of data augmentation strategies for prostate cancer detection in diffusion-weighted mri using convolutional neural networks. *arXiv preprint arXiv:2006.01693*.

Jennings, D., Hatton, B., Guo, J., Galons, J.-P., Trouard, T., Raghunand, N., ... Gillies, R. (2002, 01). Early response of prostate carcinoma xenografts to docetaxel chemotherapy monitored with diffusion mri. *Neoplasia : An International Journal for Oncology Research*, *4*. doi: 10.1038/sj.neo.7900225

Kele, P. G., & van der Jagt, E. J. (2010). Diffusion weighted imaging in the liver. *World journal of gastroenterology: WJG*, *16*(13), 1567.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, *3361*(10), 1995.

Lee, J.-G., Jun, S., Cho, Y.-W., Lee, H., Kim, G. B., Seo, J. B., & Kim, N. (2017). Deep learning in medical imaging: general overview. *Korean journal of radiology*, *18*(4), 570.

Lind, M. (2017). *Automatic segmentation of knee cartilage using quantitative mri data.*

McCarthy, P. (2021, April). *Fsleyes.* Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.4704476` doi: 10.5281/zenodo.4704476

Murphy, P., Hooker, J., Ang, B., Wolfson, T., Gamst, A., Bydder, M., ... others (2015). Associations between histologic features of nonalcoholic fatty liver disease (nafld) and quantitative diffusion-weighted mri measurements in adults. *Journal of Magnetic Resonance Imaging*, *41*(6), 1629–1638.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. Retrieved from `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`

Patil, D. D., & Deore, S. G. (2013). Medical image segmentation: a review. *International Journal of Computer Science and Mobile Computing*, *2*(1), 22–27.

Perone, C. S., cclauss, Saravia, E., Ballester, P. L., & MohitTare. (2018, November). *perone/medicaltorch: Release v0.2.* Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.1495335` doi: 10.5281/zenodo.1495335

Pierpaoli, C. (2010). *Quantitative brain mri.* LWW.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention* (pp. 234–241).

Safdar, M. F., Alkobaisi, S. S., & Zahra, F. T. (2020). A comparative analysis of data augmentation approaches for magnetic resonance imaging (mri) scan images of brain tumor. *Acta informatica medica*, *28*(1), 29.

Saood, A., & Hatem, I. (2021). Covid-19 lung ct image segmentation using deep learning methods: U-net versus segnet. *BMC Medical Imaging*, *21*(1), 1–10.

Sengun, K., Cetin, Y., Guzel, M., Can, S., & Bostanci, E. (2021). Automatic liver segmentation from ct images using deep learning algorithms: A comparative study. *arXiv preprint arXiv:2101.09987*.

Shenoy-Bhangle, A., Baliyan, V., Kordbacheh, H., Guimaraes, A. R., & Kambadakone, A. (2017). Diffusion weighted magnetic resonance imaging of liver: Principles, clinical applications and recent updates. *World journal of hepatology*, *9*(26), 1081.

Smith, L. N. (2018). A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *CoRR*, *abs/1803.09820*. Retrieved from `http://arxiv.org/abs/1803.09820`

Sørensen, T. J. (1948). *A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons* (Vol. 5). Munksgaard Copenhagen.

Stacke, K. (2016). *Automatic brain segmentation into substructures using quantitative mri.*

Student. (1908). The probable error of a mean. *Biometrika*, 1–25.

Talbar, S., et al. (2020). Some studies on automatic liver segmentation.

Troelstra, M. A., Witjes, J. J., van Dijk, A.-M., Mak, A. L., Gurney-Champion, O., Runge, J. H., ... others (2021). Assessment of imaging modalities against liver biopsy in nonalcoholic

fatty liver disease: The amsterdam nafld-nash cohort. *Journal of Magnetic Resonance Imaging*.

Wang, K., Mamidipalli, A., Retson, T., Bahrami, N., Hasenstab, K., Blansit, K., ... others (2019). Automated ct and mri liver segmentation and biometry using a generalized convolutional neural network. *Radiology: Artificial Intelligence*, *1*(2), 180022.

Wardhana, G., Naghibi, H., Sirmacek, B., & Abayazid, M. (2021). Toward reliable automatic liver and tumor segmentation using convolutional neural network based on 2.5 d models. *International journal of computer assisted radiology and surgery*, *16*(1), 41–51.

APPENDIX A

A Tool for Automatic Liver Segmentation of qMRI Images

The code that was developed for this project is available on GitHub (`https://github.com/dilaratank/BScThesis-AutoLiverSeg`). The code includes the conventional U-Net and SegNet models and the additional implementations, a training pipeline (in Pytorch), running scripts, visualization notebooks, and a tool for automatic liver segmentation. The tool was developed with the best performing model in fold 1 (see Table 4.1) and the post-processing method described in section 3.2.4. The tool does not have a graphical user interface but can be used on a Linux command line.